IBM z/VSE

Administration

Version 3 Release 1

SC33-8224-00

IBM z/VSE

Administration

Version 3 Release 1

SC33-8224-00

Note!

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

First Edition (March 2005)

This edition applies to Version 3 Release 1 of IBM z/Virtual Storage Extended (z/VSE), Program Number 5609-ZVS, and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the addresses given below.

A form for readers' comments is provided at the back of this publication. If the form has been removed, address your comments to:

IBM Deutschland Entwicklung GmbH
Department 3248
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany

You may also send your comments by FAX or via the Internet:

Internet: s390id@de.ibm.com
FAX (Germany): 07031-16-3456
FAX (other countries): (+49)+7031-16-3456

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. 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 of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785, U.S.A.

Any pointers in this publication to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement. IBM accepts no responsibility for the content or use of non-IBM Web sites specifically mentioned in this publication or accessed through an IBM Web site that is mentioned in this publication.

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 Deutschland GmbH
Department 0790
Pascalstr. 100
70569 Stuttgart
Germany
```

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

Programming Interface Information

This manual also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of z/VSE. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

Programming Interface information

End of Programming Interface information

Trademarks and Service Marks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

ACF/VTAM	ESA/390	System/390
Advanced Function Printing	IBM	S/390
AFP	Language Environment	Virtual Machine/ Enterprise Systems Architecture
C/370	MQSeries	VM/ESA
CICS	Netfinity	VSE/ESA
CICS/VSE	NetView	VTAM
DB2	OS/2	WebSphere
DB2 Connect	OS/390	xSeries
DB2 Server for VSE	OS/400	z/OS
DFSORT	Print Service Facilities	zSeries
ECKD	PSF	z/VM
Enterprise Systems Architecture/390	System/370	z/VSE

Java, Java Beans, and JavaScript are trademarks or registered trademarks of Sun Microsystems, Inc..

Microsoft, Windows, the Windows 95 logo, Windows NT and Windows XP, are trademarks or registered trademarks of Microsoft Corporation.

Visual Basic is a registered trademark of Microsoft Corporation.

Lotus, 1-2-3, Domino, and Notes are trademarks or registered trademarks of the IBM-Lotus Corporation.

SET and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, and service names, may be trademarks or service marks of others.

About This Book

z/VSE is the successor to IBM's VSE/ESA product. Many products and functions supported on z/VSE may continue to use VSE/ESA in their names.

z/VSE can execute in 31-bit mode only. It does not implement z/Architecture, and specifically does not implement 64-bit mode capabilities.

z/VSE is designed to exploit select features of IBM eServer zSeries hardware.

This manual mainly describes how to perform resource definition tasks for IBM z/VSE Version 3, Release 1. z/VSE belongs to the Enterprise Systems Architecture/390 (ESA/390) operating systems designed for the System/390 environment.

Resource definition tasks are tasks which define the characteristics of your z/VSE system. System resources include items such as startup procedures, user profiles, files, libraries, and devices installed. This manual describes how to define and modify such resources.

Who Should Use This Book

This manual is mainly intended for the system administrator or persons performing resource definition tasks. Some of the information provided concerns programmers as well.

How to Use This Book

For many resource definition tasks, z/VSE provides dialogs via the Interactive Interface. The manual shows for the model system administrator (SYSA) how to access a dialog for a particular task. It shows (in boxes) the fast path and the default synonym, if available. In the synonym box, space is left to allow you to add your own synonym. For those tasks for which z/VSE provides skeletons, the manual describes the skeletons and shows when and how to use them.

Where to Find More Information

An overview of z/VSE 3.1 enhancements and changes is provided in the *z/VSE Release Guide*.

z/VSE Home Page

z/VSE has a home page on the World Wide Web, which offers up-to-date information about VSE-related products and services, new z/VSE functions, and other items of interest to VSE users.

You can find the z/VSE home page at:

`http://www.ibm.com/servers/eserver/zseries/zvse/`

Summary of Changes

- **z/VSE** is the successor to **VSE/ESA**. However, the names of many features and programs related to z/VSE remain unchanged (such as IBM Language Environment for VSE/ESA, IBM COBOL for VSE/ESA, or TCP/IP for VSE/ESA).
- A large Environment C has been provided which has a maximum virtual size (VSIZE) of 2 GB. See "Skeleton SKALLOCC" on page 35.
- Step-by-step information is included of how to configure your z/VSE system to use SCSI disks. See Chapter 5, "Configuring Your System to Use SCSI Disks," on page 81.
- Details of how to specify a lock file for use with FCP-attached FBA-SCSI disks have been provided. See "Changing Startup When Lock File Is Stored On SCSI DASD" on page 28.
- Information is included describing how z/VSE stores the password history for each user. See "How the Password History Is Stored" on page 126.
- Information is included describing how to use the IESIRCVT program to change the required minimum length of user passwords, and the number of invalid sign-on attempts a user can make before his/her password is revoked. See "Changing the Minimum Password Length and Revoke Details" on page 127.
- Information is included describing the Virtual Tape Support, which enable you to read from or write to a virtual tape in the same way as if it were a physical tape. See Chapter 13, "Using Virtual Tapes," on page 265.
- Information is included describing the support for the IBM TotalStorage ESS 3494 Tape Library, which allows the IBM 3494 tape library to be supported via the S/390 channel command interface. As a result, there is no longer a requirement to use the XPCC/APPC communication protocol. See Chapter 14, "Implementing Native 3494 Tape Library Support," on page 281.
- Information is included describing the support for the IBM TotalStorage ESS *FlashCopy* feature, which enables you to take a copy of a total volume, a range of cylinders, or a non-VSAM file. See Chapter 15, "Performing a FlashCopy Using the IXFP Command," on page 287.
- The manual has been updated to include z/VSE's support for the DB2 Server for VSE Version 7 Release 4. The skeletons containing statements that reference the DB2 Server for VSE have been changed accordingly.

What is New With z/VSE 3.1?

For a summary of *all* the items that have been introduced with z/VSE 3.1, refer to the *z/VSE Release Guide*, SC33-8220.

Chapter 1. Using the Interactive Interface and Skeletons

The z/VSE Interactive Interface makes it easier for you to interactively use the facilities of z/VSE and its components. You select the task you want to perform from selection panels. A dialog requests input from you to complete the specific task. Some functions are not supported by dialogs but by skeletons. To perform a task with a skeleton, you edit (change) the skeleton and submit it for processing.

z/VSE Profiles

z/VSE provides model user profiles. A user profile defines a user to the z/VSE System. It includes a user ID and password which you use to sign on to the system. The profile defines what is invoked after you sign on. The model profiles reflect different levels of authorization. The user IDs and corresponding passwords provided with z/VSE are shown below.

Table 1. Model User Profiles of z/VSE

User ID	Password	Function
SYSA	SYSA	Model system administrator
PROG	PROG	Model programmer
OPER	OPER	Model operator
POST	BASE	User to complete initial installation
CICSUSER	CICSUS	CICS default User
DBDCCICS	DBDCCI	CICS partition user (F2)
PRODCICS	PRODCI	CICS partition user (F8)
CNSL	CNSL	Default CICS user with administrator authority for the internal master console and other internal consoles
FORSEC	FORSEC	Model system administrator (without VSE/ICCF)
$SRV	$SRV	Model for problem determination
VCSRV	VCSRV	Connector Server/Virtual Tape Data Handler partition user

You can use the first three profiles as models to define your own user IDs for an administrator, programmer, or operator. It is recommended that you do not change or delete the authorizations of SYSA, PROG, OPER, or FORSEC. They can be affected when you perform a Fast Service Upgrade. After initial installation, you should define and use your own user IDs. Use the *Maintain User Profiles* dialog for defining user IDs.

CICSUSER is a default user ID required for CICS Transaction Server startup. It performs security checks for terminal users that are not signed-on. DBDCCICS and PRODCICS are partition user IDs required for CICS Transaction Server startup. CNSL is the default user used with the master console and other internal consoles (such as REXX consoles). These user IDs do not have initial selection panels and can therefore not be used for sign-on.

Note: Do NOT delete the CICSUSER or CNSL user IDs.

FORSEC is a model user ID and password for system startup with security (access control) active. This user ID is relevant only, if you have selected SECURITY=YES during initial installation of z/VSE or if you want to activate it later. For further details on security, refer to Chapter 7, "Protecting Resources," on page 129.

$SRV identifies a default panel hierarchy which provides access to a set of standard dialogs for problem determination. This panel hierarchy is mainly intended for IBM personnel doing remote problem determination for a user site via a data link connecting the user installation with an IBM Support Center, for example. But the $SRV panel hierarchy can also be used for local problem determination. For details about the dialogs available with $SRV, refer to the manual *z/VSE Guide for Solving Problems* under "Model User Profile for Problem Determination".

VCSRV is the partition user required for the Connector Server and the Virtual Tape Data Handler startup. It does not have an initial selection panel and can therefore not be used for sign-on.

Appendix A, "Fast Paths and Synonyms for Dialogs," on page 377 provides an overview of the dialogs available and shows which model user ID can access which dialog.

The panel hierarchies for SYSA, PROG, and OPER are shown in the foldouts at the back of the manual.

After initial installation you must change the passwords of SYSA, PROG, OPER, and $SRV during the first logon.

Types of Interactive Interface Panels

The Interactive Interface uses several types of panels:

Selection Panels

A selection panel displays up to nine options which you can select. The selections are numbered. You make your selection by entering the appropriate number at the bottom of the panel.

After you have entered the selection, either another selection panel is displayed or an application is invoked. This application may offer you a:
- Selection panel
- Data entry panel
- Function list
- Application sign-on panel

Data Entry Panels

The dialogs use data entry panels to obtain input about the task you are performing. You enter information in particular fields on the panel. For example, if you define a library, you must enter the library file name in the LIBRARY NAME field.

Function Lists

A Function List (FULIST) displays a list of items which you can process. The items can be, for example:
- Devices

- VSE/VSAM files
- VSE/ICCF library members
- VSE/POWER queue entries

Such a panel also displays options you can use to process these items. Options could be:

- Alter
- Show
- Print
- Copy
- Delete

Each option corresponds to a particular number. When using a FULIST, you simply enter the number of the option next to the item you want to process.

Help Panels

On most panels you can press PF1 to display a HELP panel. This provides additional information about the FULIST, selection panel, or data entry panel and the task you are performing.

Sometimes the system displays a message on your panel for which specific help information is available. For example, if you have entered incorrect data, the message informs you of the error. If you then press PF1 in the case of CICS panels, the HELP panel explains the error and how you can correct it. Some HELP panels display a list of topics which you can select for more information.

Using the Fast Path Facility

The Interactive Interface has a “fast path” facility which allows you to go directly to a dialog without going through the entire panel hierarchy to reach the dialog. You will use the fast path facility after having some experience and knowing the selection numbers required for a task.

To use a fast path, you enter all the numbers (on one selection panel) of the selections you would enter on the individual panels in the hierarchy. An example is given below to show you how a fast path works. After signing on with an administrator user ID, the system displays the *z/VSE Function Selection* panel. Suppose you want to access the *Maintain User Profiles* dialog. To reach the dialog by going through the entire hierarchy, you would do the following:

In the *z/VSE Function Selection* panel, select:

2 (Resource Definition)

In the *Resource Definition* panel, select:

1 (User Interface Tailoring)

In the *User Interface Tailoring* panel, select:

1 (Maintain User Profiles)

By using the fast path, you can go directly to the *Maintain User Profiles* dialog. In the *z/VSE Function Selection* panel, enter:

211

A fast path may be entered on any selection panel. It represents the selections starting from the current panel. If a fast path is preceded by the character ’=’, selection begins at the *z/VSE Function Selection* panel. Note that if you use the fast path facility to access a dialog, PF3 returns to the point where the fast path started.

Appendix A, "Fast Paths and Synonyms for Dialogs," on page 377 shows the fast paths that can be used to access dialogs.

Using the Synonym Function

This function allows you to use a private synonym for selecting a panel. For example, instead of specifying the fast path **211** for the *Maintain User Profiles* dialog you can also enter a word which you may remember better. **upm** is the default synonym provided by z/VSE for the *Maintain User Profiles* dialog.

Synonym models are shipped for the model users SYSA, PROG, and OPER. Refer to Appendix A, "Fast Paths and Synonyms for Dialogs," on page 377 for the synonyms provided for these users. With the *Maintain Synonym* dialog, users can change their synonyms and create new ones. Refer to "Maintaining Synonyms" on page 125 for a description of the dialog. Note that if you use a synonym to access a dialog, PF3 returns to the initial function selection panel.

Note: For the dialogs described in this book, you will find two boxes showing the related fast path and the default synonym, if available. The synonym box has space left to allow you to add the synonym you created for yourself.

Signing on to the Interactive Interface

In order to use the Interactive Interface, you have to *sign on*. The sign-on procedure identifies you to the system and accesses the Interactive Interface. Before you can sign on, you need a user ID and password. The system administrator is responsible for creating user IDs.

The user ID is a 4 - 8 character name that identifies a user to the system. The password is a 3 - 8 character confidential code associated with the user ID. You sign on to the Interactive Interface from the *z/VSE Online* panel by entering your user ID and password. An example of the panel is shown in Figure 1 on page 5. The password is **not** displayed on the panel.

```
USER-ID....      xxxxxxxx
PASSWORD...      yyyyyyyy
```

The system checks the user ID and password. If they are correct, it accesses the selection panel or application which is defined for that particular user ID. "Password Expiration" on page 126 gives details on how to change a password.

Using Program Function (PF) Keys

The Interactive Interface supports Program Function (PF) keys to perform various functions. PF keys and the function they represent are displayed at the bottom of each panel.

Note: Some keyboards use 'F keys' (for example, F9) instead of PF keys. The function, however, is the same. This book uses the name 'PF' key.

Your terminal has either 10, 12 or 24 PF keys depending on the model of the terminal. If you have 24 keys, PF13 - PF24 correspond to the same functions as

PF1 - PF12 within a VSE environment. This may not be true for some applications which may use all 24 PF keys.

```
IESADMSO1                         z/VSE ONLINE
    5609-ZVS and Other Materials (C) Copyright IBM Corp. 2004 and other dates



                                ++
                               ++    VV    VV    SSSSS    EEEEEEE
                zzzzzz        ++     VV    VV   SSSSSSS   EEEEEEE
                zzzzz        ++      VV    VV  SS         EE
                 zz         ++       VV    VV   SSSSSS    EEEEEE
                zz         ++        VV    VV    SSSSSS   EEEEEE
                zzzzzz    ++          VV  VV         SS  EE
               zzzzzzz   ++            VVVV     SSSSSSS   EEEEEEE
                                        VV       SSSSS    EEEEEEE

        Your terminal is A000 and its name in the network is D3000001
        Today is 08/06/2004   To sign on to DBDCCICS  --  enter your:

       USER-ID........ ________   The name by which the system knows you.
       PASSWORD.......            Your personal access code.

PF1=HELP       2=TUTORIAL   3=TO VM      4=REMOTE APPLICATIONS     6=ESCAPE(U)
                            9=Escape(m) 10=NEW PASSWORD
```

Figure 1. z/VSE Online Panel

On a PC keyboard with only 10 PF keys, the following applies:

- PF9 corresponds to the PF11 key when pressed simultaneously with the shift key.
- PF10 corresponds to the PF12 key when pressed simultaneously with the shift key.

Some PF keys used by the Interactive Interface have the same function from every panel that uses them. Other PF key functions differ for different dialogs. Each panel shows the PF keys you can use and the functions to which they correspond. When you use a PF key, review the panel you are working with to know which function the PF key represents.

The following table shows the PF keys that have the same meaning on all panels of the Interactive Interface:

Table 2. Standard PF Key Usage

PF KEY	NAME	ACTION
1	HELP	Help information.
3	END	Quit and go back one level.
4	RETURN	Quit and go to top panel.
7	BACKWARD	Scroll (move) to previous page.
8	FORWARD	Scroll (move) to next page.

PF keys 2, 5, 6, 9, 10, 11, and 12 are used for different functions, but not for a function shown in Table 2.

Using Skeletons

Overview

z/VSE provides skeletons to help you complete a number of tasks. A skeleton is a member in a VSE/ICCF library. You use it to create a job which completes a task. It contains variables and parameters which you change to reflect your requirements. After you make the changes, you submit the completed skeleton (job) to the system for processing.

z/VSE ships most skeletons in VSE/ICCF library 59. They are intended to be used more than once. Therefore, you should **copy** them to another library (usually your VSE/ICCF primary library) before you edit and change them. Keep the original skeleton in its library for future use.

A skeleton is replaced if it is affected by service such as FSU (Fast Service Upgrade) or when applying a PTF. Therefore, you should make sure you copy it to another library and only change the copied member.

Copying Skeletons

The Interactive Interface provides the *Program Development Library* dialog which you can use to copy VSE/ICCF members between libraries. This section describes how you use the dialog to copy skeletons from the library in which they reside to your primary library.

When you access the *Program Development Library* dialog, the panel displays your default primary library in the PRIMARY field. This is the library to which you copy a skeleton. Access the library where the skeleton resides as the **secondary** library.

```
SECONDARY .... 59__      (Enter library number; 59, for example)
PREFIX ....... _______   (Optionally, enter a prefix for skeleton
                          names)
OPTION ....... 2         (Enter 2 for secondary library)
```

Note: Library member names (as for skeletons) consist of up to eight characters. The PREFIX field accepts up to seven characters.

The *Secondary Library* panel is displayed after pressing ENTER. Locate the skeleton name. Copy the skeleton to your primary library by entering **4** in the OPT column. You must specify a member name in the NEW NAME column. Enter the character "=" if the new name is to be the same as the original name.

Press **PF3** to return to the *Program Development Library* panel. Access your VSE/ICCF primary library to edit the copied skeleton.

Chapter 2. Tailoring IPL and System Startup

Related Information

Before you modify the procedures for IPL (initial program load) and system startup you should be familiar with the information provided in the chapter "System Organization and Concepts" of the *z/VSE Planning* manual.

You should also be familiar with the IPL and startup information provided in the manual *z/VSE Guide to System Functions* under "Starting the System" which provides details on topics such as the following:

- The ASI master procedure ($ASIPROC).
- Establishing the communication device for IPL.
- Interrupt IPL processing for modifications.
- Loading phases into the SVA.

The description of IPL and system startup in this chapter follows the sequence (and uses the names) of a z/VSE system as shipped by IBM.

Note on Terminology

The process of IPL and system startup is also referred to as ASI (automated system initialization). In this context, startup procedures and functions are also referred to as ASI procedures and functions. This chapter uses the term ASI only if technical or terminology reasons make it advisable to do so.

Initiating System Startup

Before starting up z/VSE, the operator has to perform IML (Initial Microprogram Load) for hardware (processor) initialization.

The operator can then initiate startup for z/VSE by performing IPL. After the programs required for IPL have been loaded the appropriate IPL procedure will be processed. An IPL procedure defines specific system parameters needed during IPL processing.

The name of the IPL procedure used during initial installation depends on the disk device type on which z/VSE is to reside (DOSRES). For example, for a system residing on an:

- IBM 3390 disk device, z/VSE selects IPL procedure $IPLE90 for initial installation.
- IBM FCP-attached FBA-SCSI disk device, z/VSE selects IPL procedure $IPLEGF for initial installation.

"Appendix A" of the *z/VSE Planning* manual shows the contents of such an IPL procedure and the names of the IPL procedures shipped with z/VSE.

During initial installation, the IPL procedure is modified according to the customer's environment and is renamed to **$IPLESA**. In a running system (after installation), the IPL procedure appears under this name, for example, when working with the *Tailor IPL Procedure* dialog. You can use this dialog to add or delete an IPL procedure or modify its values.

Using a $ASIPROC Procedure

When the operator initiates IPL, z/VSE does one of the following to get the correct IPL and JCL procedure names for startup:

1. Retrieves the names from $ASIPROC, if a $ASIPROC master procedure exists. For a detailed description of creating a $ASIPROC master procedure refer to the manual *z/VSE Guide to System Functions* under "The ASI Master Procedure". As shipped, z/VSE includes a $ASIPROC for initial installation only.
2. Uses the default names: $IPLESA and $$JCL.
3. In addition, z/VSE allows the operator to interrupt startup processing. The operator can then enter the procedure names to be used by the system. This is described in detail in the manual *z/VSE Guide to System Functions* under "Interrupt and Restart the IPL Process".

Tailoring the IPL Procedure

When you tailor an IPL procedure, be aware of the predefined disk layouts used by z/VSE and shown in the manual *z/VSE Planning* under "z/VSE Disk Layouts". There you can find the location of all the system files and libraries used by z/VSE and the free space still available on DOSRES and SYSWK1. *Do not use the areas defined as reserved*.

Be aware that:

- Predefined environment B (a medium environment) contains an enlarged page data set of 264 MB (compared to 150 MB for predefined small environment A). The free space is reduced accordingly.
- Predefined environment C (a large environment) contains an enlarged page data set of 2 GB. The free space is reduced accordingly.

To access the *Tailor IPL Procedure* dialog, start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
4 (Hardware Configuration and IPL)
3 (Tailor IPL Procedure)

```
Administrator
Fast Path:  243
```

```
Synonym
Default:    ____    Yours:
```

The panel displayed lists the IPL procedures defined for your system. **ADD**, **ALTER**, and **DELETE** are the options you can select. Enter the option number in the OPT column next to the procedure you want to process.

Adding or Altering an IPL Procedure

If you add a procedure, the procedure you select is used as a model. The values defined for the model are used as defaults for the new procedure. You are requested for a name of the new IPL procedure. This is the name you must include in $ASIPROC or the operator must enter it during IPL.

IPL Parameters You Can Modify

If you add or alter an IPL procedure, you can modify the IPL parameters listed below. Refer to the manual *z/VSE System Control Statements* for a detailed description of the IPL commands and parameters or use the help text provided by the dialog via PF1.

- **Supervisor**

  Used to modify parameters such as the address of the IPL console and virtual storage options (VSIZE, VIO, VPOOL). Figure 2 on page 11 shows the panel layout and the parameters that can be specified.

- **SYS**

  Used to modify various system parameters displayed on two panels:

  Panel TAS$ICM1

  ```
  BUFLD
  CHANQ
  DASDFP
  SUBLIB
  VMCF
  SEC
  ESM
  SERVPART
  ```

  For details on the security parameters SEC, ESM, and SERVPART, refer to "Using the Tailor IPL Procedure Dialog to Modify Security Parameters" on page 133.

  Panel TAS$ICMA

  ```
  BUFSIZE
  NPARTS
  PASIZE
  RSIZE
  SDSIZE
  SPSIZE
  ATL
  ```

  If your processor allows automatic timer-controlled IPL, set the BUFLD parameter to IGNORE if you use this function. Refer to the description of the SYS command in the manual *z/VSE System Control Statements* under "SYS" for further details.

  The automatic IPL function is closely related to the automatic timer-controlled power-on function of a processor. If these hardware functions are available, your processor provides panels for using them.

- **DLA**

  Used to modify the label area definition.

- **DPD**

  Used to modify page data set definitions. Be aware that:
  - Predefined environment A contains a page data set of 150 MB.
  - Predefined environment B contains a page data set of 264 MB.
  - Predefined environment C contains a page data set of 2 GB.

  Note that:

  - A DPD specification is not allowed if you define a system without a page data set. Refer to "Page Data Set Considerations" on page 11 for further details.
  - You must ensure that no overlap occurs with other files when enlarging or relocating the page data set extents.

  Refer to "Page Data Set Considerations" on page 11 for further page data set details.

- **DLF**

  Used to modify the lock file (also known as cross-system communication file) definition. This file is required when sharing disk devices (DASD sharing) among VSE systems.
- **DEF**

  Used to modify the definition of the physical device for the system recorder file and the hardcopy file (SYSREC) and the VSE/VSAM master catalog (SYSCAT).
- **ZONE**

  Used to modify the ZONE specifications (time difference between local time and Greenwich Mean Time).

  For further details about ZONE specifications, refer to Chapter 20, “ZONE Specifications and Daylight Saving Time,” on page 345.
- **APPC/VM**

  Used to modify VSE APPC/VM resource definitions. Such definitions are required if you run z/VSE under VM and want to enable DB2 Server for VSE & VM applications to share one or more DB2 data bases.
- **SVA**

  Used to modify parameters of the Shared Virtual Area (SVA):

  ```
  SDL ENTRIES  (max. number is 32765)
  Additional PSIZE (24-bit, 31-bit)
  Additional GETVIS (24-bit, 31-bit)
  ```

How to Add an IPL Procedure

The following steps are required if you want to create and add a new IPL procedure:

1. In the panel that lists the IPL procedures for your system enter **1** next to an existing IPL procedure. This procedure is used as a model. The values of the model are used as defaults for the new procedure.
2. The next panel requests you to enter the **name** of the new procedure (the first character must be $).
3. The subsequent panel shows the parameters that can be modified. They are listed under “IPL Parameters You Can Modify” on page 9. Enter **1** to the left of the parameter(s) you want to change. Figure 2 on page 11 shows, as an example, the panel for changing SUPERVISOR parameters.
4. A panel will be displayed for each parameter you select. Press **PF5** to save your changes.
5. On the panel displayed next, press **PF5** to create a jobstream to catalog your new or updated IPL procedure.
6. The *Job Disposition* panel is displayed. With it, you can submit the job to batch, file it in your VSE/ICCF primary library, or both.

```
 TAS$SUP1        TAILOR IPL PROCEDURE: SUPERVISOR PARAMETERS

 ENTER THE REQUIRED DATA AND PRESS PF5=PROCESS


IPL CONSOLE.................. ___        Console address used during IPL.  For
                                         valid console addresses enter a "?".
SUPERVISOR ID................ _          Last character x of the supervisor
                                         name: $$A$SUPx.
PAGE DATA SET................ _          Do you want to use a page data set?
                                         1=yes, 2 =no.
VSIZE........................ _______    Virtual address space; in M or G.

VIO.......................... _______    Virtual I/O work space; in K or M.

VPOOL........................ ______     V-pool size; in K or M.

IPL LOG OPTION............... _          Logging of IPL commands on the IPL
                                         console required? 1=yes, 2=no.

 PF1=HELP      2=REDISPLAY  3=END
```

Figure 2. Panel for Modifying Supervisor Parameters

Page Data Set Considerations

The page data set option allows to define a system without a page data set which is possible if enough real (processor) storage is available or if, in a VM environment, the VM virtual storage size is large enough to accommodate all z/VSE storage requirements.

If you specify 2 (no) for PAGE DATA SET, you get a system without a page data set. This system status is also referred to as NOPDS. For a NOPDS-system, a specification of VSIZE and the DPD parameter in the IPL procedure are not allowed.

The manual *z/VSE Planning* provides further details about NOPDS.

You may move the page data set from the system reserved space to another location and use the reserved space for your own definitions. You can release the reserved space by using the FREE key (PF6) on the DPD panel of the *Tailor IPL Procedure* dialog.

Note that starting with VSE/ESA 2.6, z/VSE checks automatically during initial installation whether the processor storage size is sufficient for a NOPDS system. If YES, no page data set is created and a NOPDS system is established.

Overview of Startup Processing

When the operator initiates system startup, first the IPL procedure, then the JCL startup procedures and jobs are processed. As shipped, z/VSE uses the default procedure names for startup.

The JCL startup procedure for the BG partition calls, right at the beginning, the startup program **DTRISTRT**. DTRISTRT uses as input the following:

- System variables stored in procedure CPUVARn (CPUVAR1 procedure shipped with z/VSE and used in this description).
- The startup mode entered by the operator (BASIC, COLD, or MINI).

  Entering a startup mode is optional.

As output, DTRISTRT generates values for system variables and updates those variables in procedure CPUVAR1.

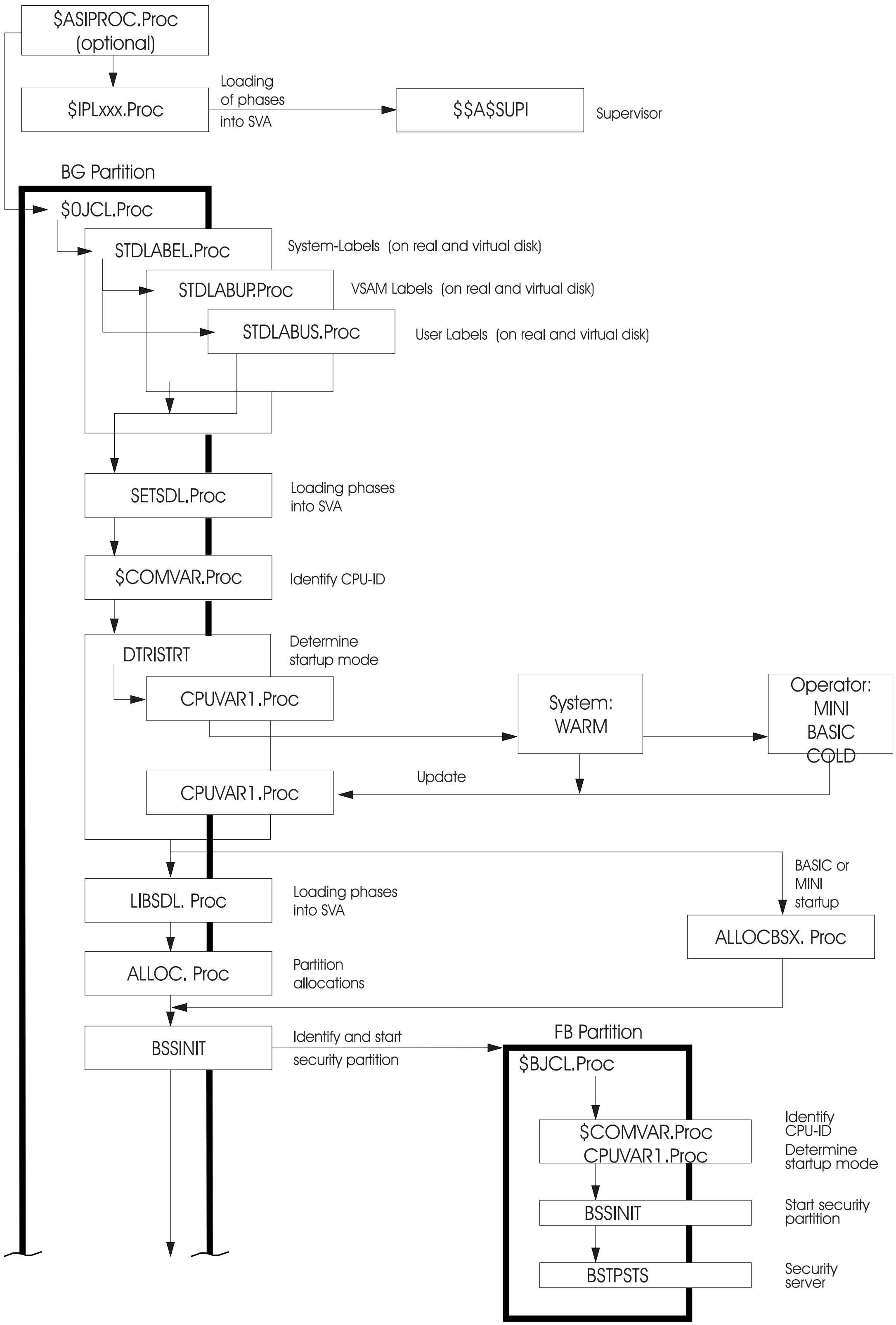
The JCL startup procedures retrieve the updated system variables from CPUVAR1. These variables control the subsequent startup process and determine the startup mode used. Refer to “JCL Startup Procedures and Jobs” on page 15 for further details.

Figure 3 on page 13 and Figure 4 on page 14 show the flow of main events during startup processing for predefined environments A, B, and C. Note that procedure POWSTRTn (where n identifies the predefined environment) also activates the dynamic partition support which is not shown in the figure.

Removed Support and z/VSE Environment C

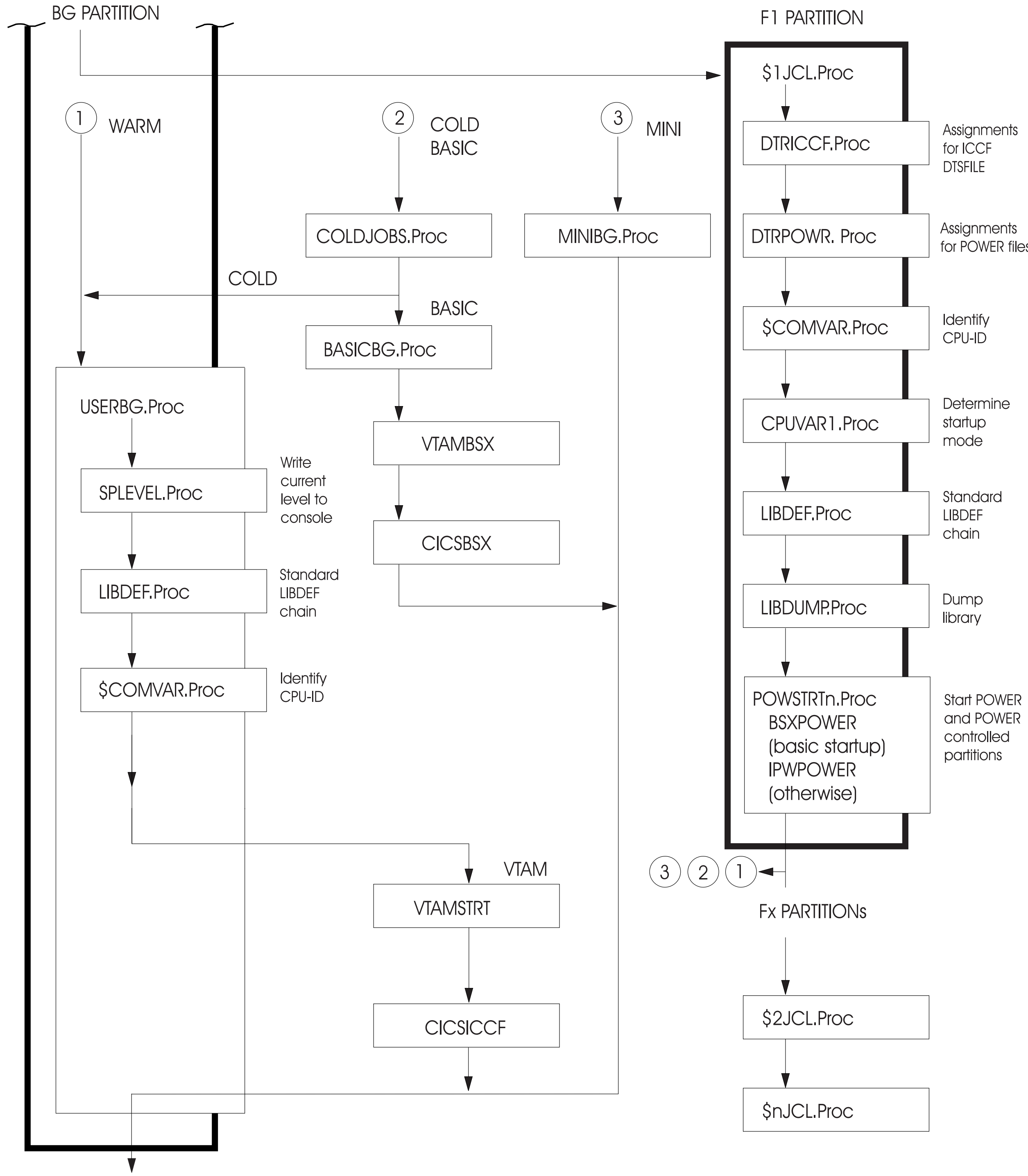
Note that:

1. From z/VSE 3.1 onwards, Environment C is introduced and is a large environment using a maximum page data set size of 2 GB. This Environment C is *not* related to the previous Environment C that was used for Unattended Node Support.
2. For details of this Environment C, refer to the manual *z/VSE Planning*, SC33-8221.



Note: As shipped, z/VSE includes a $ASIPROC for initial installation only. If you want to use $ASIPROC, you must create your own. Refer also to "Using a $ASIPROC Procedure" on page 8.

Figure 3. z/VSE Startup Sequence for an Unmodified System (Part 1)



Note: The sequence of events shown in the F1 partition is for WARM startup and continues at entry point 1. A COLD or BASIC startup continues at entry point 2, a MINI startup at entry point 3, after the corresponding sequence in F1 has completed.

Figure 4. z/VSE Startup Sequence for an Unmodified System (Part 2)

JCL Startup Procedures and Jobs

After the successful processing of the IPL procedure, the JCL startup procedure for the BG partition **$0JCL** is called. The 0 in the second position indicates the BG partition. For the naming convention of JCL startup procedures refer to "Appendix A" in the *z/VSE Planning* manual.

Following is a list of actions performed after $0JCL gets control. The list describes the flow of events (for WARM and RECOVERY startup) as shown in Figure 3 on page 13 and Figure 4 on page 14.

1. $0JCL calls procedure **STDLABEL** to write system file labels into the label information area. STDLABEL is called twice, once for writing the labels to disk, and once for writing them to the virtual disk. STDLABEL calls label procedures **STDLABUP** (VSE/VSAM labels) and **STDLABUS** (user labels).

   The dialogs for VSE/VSAM objects automatically update STDLABUP. STDLABUS, however, must be updated for user files created in space not managed by VSE/VSAM by using skeleton STDLABUS.
2. Once standard labels are written, $0JCL calls procedure **SETSDL**. This procedure writes phase names into the system directory list (SDL) and optionally loads selected phases into the SVA.
3. After SETSDL completes, $0JCL calls procedure **$COMVAR** to identify the CPU where startup is to be performed and the SETPARM procedure to be used. For a single CPU system, the name of the SETPARM procedure shipped with z/VSE is **CPUVAR1**.
4. $0JCL then calls the startup program **DTRISTRT**. DTRISTRT decides which startup mode is to be used. It bases its decision on the information stored in CPUVAR1 and the startup mode requested by the operator (if any). DTRISTRT also decides about BASIC ($$JCLBSX) and MINI ($$JCLMIN) startup. You can modify the startup only, if DTRISTRT is included in $0JCL.
5. When DTRISTRT completes, $0JCL calls procedure **LIBSDL** to load phases for VTAM, CICS, REXX, High Level Assembler, LE/VSE, the VSE C Language Run-Time Support, as well as connector-related programs into the SVA.
6. Next, $0JCL calls procedure **ALLOC** for allocating static partitions.

   If startup mode is BASIC or MINI, procedure ALLOCBSX is called instead.
7. The security server partition is started by job SECSERV according to the return code of program BSSINIT. The related variable in CPUVARn is set. As shipped, the security server runs in the FB partition. The security server must run in a partition that is not under the control of VSE/POWER. BSSINIT is executed again to identify the security server partition. Based on the result, program BSTPSTS is executed, which is the actual security server program. The security server is stopped after VSE/POWER is terminated with the PEND command.
8. Once the partitions are allocated, partition F1 is started and $1JCL activates VSE/POWER.
9. $1JCL calls the following procedures:
   a. **DTRICCF** for assigning the VSE/ICCF DTSFILE.
   b. **DTRPOWER** for assigning VSE/POWER files.
   c. **$COMVAR** to identify the CPU where startup is to be performed and the SETPARM procedure to be used.
   d. **CPUVAR1** for retrieving environment information.
   e. **LIBDEF** for defining standard LIBDEF search chains for libraries and sublibraries.

f. **LIBDUMP** for assigning the dump library for VSE/POWER.

g. **POWSTRTn** to start all VSE/POWER-controlled partitions allocated (except F1, which is already running). The following VSE/POWER startup procedures are provided:
   POWSTRTA (predefined environment A)
   POWSTRTB (predefined environment B)
   POWSTRTC (predefined environment C)

   When a partition has been started via POWSTRTn, the JCL ASI procedure for that partition is processed: $2JCL, $3JCL, and so on.

   Note the following:
   - If startup mode is COLD, the VSE/POWER queues are also formatted.
   - If startup mode is BASIC, procedure $1JCLBSX
     – brings up the F1 partition
     – formats temporary VSE/POWER queues on SYSWK1, and
     – starts partitions BG, F2, F3, F4, and F5.
   - If startup mode is MINI, $1JCLMIN initializes a two-partition system (BG and F1).

10. After partition startup completes, $0JCL resumes processing. The BG partition is now under control of VSE/POWER.

    Depending on the startup mode, the following procedures are called:

```
Mode:   Procedure(s) called:

WARM    USERBG
COLD    COLDJOBS, USERBG
BASIC   COLDJOBS, BASICBG
MINI    MINIBG
```

    These procedures determine the remaining startup activities.

USERBG

USERBG is processed if startup mode is WARM or COLD. It calls the procedures SPLEVEL, LIBDEF, $COMVAR, and CPUVAR1, and releases the startup jobs VTAMSTRT and CICSICCF.

COLDJOBS

COLDJOBS is processed if startup mode is COLD or BASIC. The procedure reloads jobs provided by z/VSE into the VSE/POWER reader queue.

BASICBG

BASICBG is processed if startup mode is BASIC. It loads the startup jobs VTAMBSX and CICSBSX into the VSE/POWER reader queue.

Together with COLDJOBS, BASICBG completes the building of a basic system without user modifications. A basic system helps you recover from a problem caused, for example, by your own modifications.

MINIBG

MINIBG is processed if startup mode is MINI. It completes the building of a two partition system, BG and F1 (VSE/POWER). You can use such a system for library maintenance, for example.

Procedures CPUVARn and $COMVAR

One or more CPUVARn procedures are the memory of the startup process. A CPUVARn procedure consists of JCL SETPARM statements which contain system variables. These variables describe each partition's status. Each CPU requires its own CPUVAR**n** procedure, where **n** identifies the CPU. For a single CPU system, z/VSE provides procedure CPUVAR1.

Procedure $COMVAR identifies the CPU number and the related CPUVARn procedure of the CPU on which the startup is to be performed.

Refer to "CPUVARn and Related Startup Processing" on page 20 for an example of a CPUVARn procedure.

Considerations for Tailoring System Startup

Plan for extensive testing if you modify the system startup. Before you implement your changes, ensure that they are error free and that system startup still works correctly.

Procedures and Jobs You Should Not Change

There are a number of startup procedures and jobs which you should not change. These procedures and jobs control the startup modes BASIC and MINI. Both startup modes enable you to bring up z/VSE even in case of problems. It is, therefore, essential that you can always perform a BASIC or MINI startup. The following procedures and jobs are used for a BASIC or MINI startup and should not be changed:

- $1JCLBSX
- $1JCLMIN
- $BJCLMIN
- BASICBG
- MINIBG
- ALLOCBSX
- CICSBSX
- VTAMBSX

Considerations for BASIC and MINI Startup

You should be aware of the following problems that may arise when you are using a MINI or a BASIC startup in a modified environment.

- BASIC startup

  A BASIC startup uses for partition allocations the procedure ALLOCBSX as shipped with z/VSE. If you change the virtual storage values in your IPL procedure or generate a supervisor which requires considerably more storage, the storage required by BASIC startup for partition allocation may not be available. You may then get a system with only a limited number of partitions active.
- MINI startup

  In an environment with VSE/POWER **shared spooling**, you have regenerated and modified the VSE/POWER phase IPWPOWER and probably given it a name of your own. If you select a MINI startup, the system uses the original IPWPOWER phase in which shared spooling is defined as **SHARED=NO**. To avoid a possible damage of your VSE/POWER files, you must first shutdown all the other involved VSE systems before you select a MINI startup.

Note: The VSE/POWER queue file resides in the partition GETVIS area. The partition allocation for VSE/POWER in case of a MINI startup is based on the **default** size of the VSE/POWER queue file. If you have increased the size of the queue file, it does no longer fit into the partition GETVIS area and the MINI startup does not work. If this is the case, perform a BASIC startup to get a system with the BG and F1 partition active.

Job Control Language Used

The startup procedures and jobs are written in the job control language (JCL). JCL functions such as conditional job control, symbolic parameters, and nested procedures can be used. When modifying system startup, you must adhere to the rules valid for these functions. For a detailed description of these functions, refer to the manual *z/VSE Guide to System Functions* under "Controlling Jobs".

A CPUVARn procedure consists of JCL SETPARM statements. The SETPARM statement is discussed in the manual *z/VSE System Control Statements* under "SETPARM".

Considerations for Naming Conventions

When tailoring system startup, you must plan for the names you are going to use for the changed startup procedures and jobs. The skeletons provided for startup tailoring reflect the original startup members and show the names used by z/VSE. These names reveal to you the logical structure and relationship of the startup procedures and jobs.

Using the Same Names as z/VSE

You can use the same names as z/VSE. That is, you do not change the names given in the skeletons. As a result, the changed member replaces the original member in libraries IJSYSRS.SYSLIB (and in PRD2.SAVE). This means that the original startup member is lost. Consequently, if your changes are incorrect, you will get startup problems. Although the manual *z/VSE Guide for Solving Problems* provides hints to overcome such a situation, try to avoid such a problem altogether. Use *one* of the following methods:

- Create backup copies of those startup members you are going to change. Create new ones as required but assign different names to them. After successful testing, rename them to the original z/VSE-supplied name.
- Change the skeletons (if necessary) so that your changed version is not stored in PRD2.SAVE, but only in IJSYSRS.SYSLIB. The original member is then still available in PRD2.SAVE. After successful testing, save the changed version in PRD2.SAVE as well.

Note: z/VSE uses PRD2.SAVE to save members. This is to ensure that the latest version of a member is not lost when performing an FSU (Fast Service Upgrade).

Using Your Own Naming Convention

If you want to use your own names, invent names that allow you easy identification of your own startup procedures and jobs.

For the JCL startup procedures **$nJCLnnn**, you must observe the following naming rules:

- The procedure name must always start with **$n** (where n is the partition number).

- The characters (maximum of six) following the first two ($n) characters must be the same for all JCL procedures.

So, if you choose your own eight character procedure names, you can change the last six characters. But these six characters must be the same for all JCL procedures required for a **single** startup.

The names chosen by you are not known to the system; you have two ways to define them to z/VSE:

- The operator enters them during IPL.
- You specify the names in a $ASIPROC master procedure. Refer to “Using a $ASIPROC Procedure” on page 8 for details.

CPUVARn and Related Startup Processing

SETPARM procedure CPUVARn controls startup processing together with startup program DTRISTRT and others. **CPUVAR1.PROC** is the default CPUVARn procedure shipped with z/VSE and shown in Figure 5.

```
// SETPARM XSPINIT=FIRST          * SYSTEM VARIABLES *
// SETPARM XENVNR=A
// SETPARM DASD=''
// SETPARM XDOSRES=''
// SETPARM TPMODE=''
// SETPARM TAPECUU=''
// SETPARM TAPEMD1=''
// SETPARM TAPEMD2=''
// SETPARM CPUMODE=''
// SETPARM XS=''
// SETPARM XPWCNTL=ALL
// SETPARM XCUST=NOAUTO
// SETPARM XUSEBG=B0              * PROGRAM RUNNING IN PARTITION *
// SETPARM XUSEF1=PW
// SETPARM XUSEF2=CI
// SETPARM XUSEF3=VT
// SETPARM XUSEF4=B4
// SETPARM XUSEF5=B5
// SETPARM XUSEF6=NONE
// SETPARM XUSEF7=NONE
// SETPARM XUSEF8=NONE
// SETPARM XUSEF9=NONE
// SETPARM XUSEFA=NONE
// SETPARM XUSEFB=NONE
// SETPARM XSTATBG=INACTIVE       * STATUS OF PARTITION *
// SETPARM XSTATF1=INACTIVE
// SETPARM XSTATF2=INACTIVE
// SETPARM XSTATF3=INACTIVE
// SETPARM XSTATF4=INACTIVE
// SETPARM XSTATF5=INACTIVE
// SETPARM XSTATF6=''
// SETPARM XSTATF7=''
// SETPARM XSTATF8=''
// SETPARM XSTATF9=''
// SETPARM XSTATFA=''
// SETPARM XSTATFB=''
// SETPARM XPARTB0=BG             * PARTITION PROGRAM IS RUNNING IN *
// SETPARM XPARTPW=F1
// SETPARM XPARTCI=F2
// SETPARM XPARTVT=F3
// SETPARM XPARTB4=F4
// SETPARM XPARTB5=F5
// SETPARM XSECP=FB
// SETPARM XPWMODE=COLD           * STARTUP MODE FOR KEY PROGRAMS *
// SETPARM XSTRTPW=WARM
// SETPARM XSTRTCI=''
// SETPARM XSTRTVT=''
// SETPARM XBASIC=NONE
// SETPARM XCOLD=NONE
// SETPARM XMODEBG=COLD           * PARTITION STARTUP MODE *
// SETPARM XMODEF1=COLD
// SETPARM XMODEF2=COLD
```

Figure 5. Example of a CPUVAR1 Procedure (Part 1 of 2)

```
// SETPARM XMODEF3=COLD
// SETPARM XMODEF4=COLD
// SETPARM XMODEF5=COLD
// SETPARM XMODEF6=''
// SETPARM XMODEF7=''
// SETPARM XMODEF8=COLD
// SETPARM XMODEF9=''
// SETPARM XMODEFA=''
// SETPARM XMODEFB=''
// SETPARM XAPPLF2=DBDCCICS       * CICS APPLICATION NAMES *
// SETPARM XAPPLF8=PRODCICS
// SETPARM XAPPLF4=OLDCICS
// SETPARM SSLCAUT=NO             * SSL CLIENT AUTHENTICATION *
```

Figure 5. Example of a CPUVAR1 Procedure (Part 2 of 2)

PARAMETERS:

CPUMODE (reflects the supervisor mode; X stands for ESA)

DASD (disk device type of DOSRES)

DISTRIB (distribution medium)

TAPECUU (tape address of installation tape)

TAPEMD1 (tape mode is streaming)

TAPEMD2 (tape mode is non-streaming)

TPMODE (VTAM)

XDOSRES (disk address of DOSRES)

XS (subarea number if z/VSE is an unattended node; it is included for compatibility reasons only since unattended node environments are no longer supported)

XSECP (security server partition)

VARIABLES:

- **XAPPLyy** (where yy is the partition ID: F2, F8, or F4)

  Use: Identifies the application names of the CICS systems running per default in F2 and F8 (CICS TS) and F4 (CICS/VSE). Must be changed if other partitions are used.

  Value: DBDCCICS | PRODCICS | OLDCICS

  Set: For F2 during initial installation (DBDCCICS). For F8 (PRODCICS) and F4 (OLDCICS) by user when installing the corresponding CICS system.

- **XBASIC**

  Use: Keeps request for a system BASIC startup from a z/VSE component program. This request will be processed at the next startup.

  Value: BASIC | NONE

  Initial: Program DTRISTRT sets it to NONE after processing of first startup after initial installation.

  Set: By z/VSE program or user application.

If a job or procedure requests the next startup to be a system BASIC start, the following JCL statement has to be used (assuming a CPU number of 2):

```
// EXEC DTRSETP,SIZE=AUTO,PARM='CPUVAR2;;SET XBASIC=BASIC'
```

- **XCOLD**

  Use: Keeps request for a system COLD start. This request will be processed at the next startup.

  Value: COLD | NONE

  Initial: Program DTRISTRT sets it to NONE after processing the request.

  Set: By z/VSE program or user application. For example, when extending VSE/POWER queues or installing VSE/POWER PNET.

- **XCUST**

  Use: Keeps request for automatic customization of an unattended node (it is included for compatibility reasons only since unattended nodes are no longer supported).

  Value: AUTO | NOAUTO

  Initial: NOAUTO

  Set: Via DTRSETP program during installation of the unattended node system at the service node.

- **XENVNR**

  Use: Contains the character of the predefined environment selected during initial installation. This number determines the POWSTRTn procedure to be called by the JCL procedure of the VSE/POWER partition ($1JCL).

  Value: A or B.

  Set: During initial installation.

- **XMODEyy** (where yy is a partition ID: BG, F1 to FB)

  Use: Contains the startup mode to be performed for a particular partition.

  Value: MINI | BASIC | COLD | RECOV | WARM

  Set: Program DTRISTRT decides on the startup mode for each partition and sets all variables XMODEyy accordingly.

- **XPARTzz** (where zz is the symbolic partition ID as defined in XUSEyy)

  Use: Keeps partition ID of symbolic partition. Necessary for mapping to actual partitions. For example, statement START &XPARTPW in the BG ASI procedure will be changed (if this variable contains the value F1), into the statement START F1 to start the VSE/POWER partition.

  Value: BG | F1 through FB

  Set: By program DTRISTRT, but not for a BASIC startup. For a BASIC startup, the use of the partitions is predetermined.

- **XPWCNTL**

  Use: Keeps status of partition control during installation of an unattended node system (it is included for compatibility reasons only since unattended nodes are no longer supported).

Value: ALL | NET

Initial: ALL

Set: By the system during initial installation of an unattended node system at the service node.

- **XPWMODE**

  Use: Contains the startup mode to be executed for the VSE/POWER partition.

  Value: Same as for XMODEyy above.

  Set: By program DTRISTRT. DTRISTRT decides on the startup mode for each partition and sets this variable accordingly.

- **XSECP**

  Use: Server partition as specified in the IPL SYS command.

  Value: FB | PARTITION ID

  Initial: FB

  Set: By the system during startup of BG partition.

- **XSPINIT**

  Use: Used by program DTRISTRT to keep track of how the installation of z/VSE is progressing.

  Value: FIRST | INSTALL | FINISHED

  Initial: FIRST

  Set: Is reset to INSTALL at initial installation time and set to FINISHED when initial installation is complete.

- **XSTATyy** (where yy is a partition ID: BG, F1 to FB)

  Use: Keeps status of each partition. This is used by the startup program DTRISTRT to decide between a WARM or RECOV (recovery) startup.

  Value: ACTIVE | INACTIVE | blank

  Set: By program DTRSETP: to ACTIVE when startup job begins, to INACTIVE when startup job ends.

- **XSTRTzz** (where zz is the symbolic partition ID as defined in XUSEyy)

  Use: Keeps request for a COLD startup from a z/VSE program or user application. This request will be processed at the next startup.

  Value: COLD | NONE

  Set: By z/VSE programs or user applications. Program DTRISTRT sets it to NONE after processing the request.

- **XUSEyy** (where yy is a partition ID: BG, F1 to FB)

  Use: Indicates use of partition. The value is used to construct the name of related variables. For example, VT to build the names XSTRTVT and XPARTVT.

Value: NONE, or two alphanumeric characters that represent a symbolic partition ID. For example: B0, PW, CI, VT, C2, B4, and so on.

Note: If you want to exclude a partition from startup processing, you must specify a value of NONE.

The following values are reserved:

B0 = Background partition

PW = VSE/POWER

CI = CICS with VSE/ICCF

VT = VTAM

Initial: Standard use of partitions:

XUSEBG = B0

XUSEF1 = PW

XUSEF2 = CI

XUSEF3 = VT

XUSEF4 = B4

XUSEF5 = B5

Set: By **customer** when the use of a partition changes.

Note that it is permitted to set several XUSEyy variables to the same value. For example, to B4 for a batch partition. The XUSEyy variables must be present for each partition generated in the supervisor.

- **SSLCAUT**

  YES Implement client authentication. For details, refer to the chapter "Configuring for Client Authentication" in the manual *z/VSE e-business Connectors User's Guide*, SC33-8231.

  NO Do not implement client authentication.

Startup Program DTRISTRT

DTRISTRT is activated by the JCL startup procedure for the BG partition: $0JCL. At that time, the system variables of CPUVARn still reflect the status of the latest shutdown or startup (if shutdown was not performed or not successful).

DTRISTRT analyzes the information stored in CPUVARn for each partition before making a decision about the startup mode. The JCL startup procedures retrieve the variables from CPUVARn to initiate startup with the proper startup mode.

In addition, DTRISTRT issues messages that allow the operator to intervene and request startup modes **MINI**, **BASIC**, or **COLD**, but not WARM or RECOV (recovery). Intervention is possible if it was requested using the:
- The IPL load parameter.
- Command **MSG BG** within a three-second time limit, after message IESI0211 was issued by z/VSE during the startup of partition BG.

Refer to the manual *z/VSE Guide to System Functions* under "Interrupt IPL Processing for Modifications" for details.

For the parameters and the syntax to be used when calling DTRISTRT, refer to "Tracing Startup Processing" on page 26. There, you also find the return codes issued by DTRISTRT.

Security Initialization During Startup

The z/VSE predefined environments use FB as security server partition. It is recommended not to switch to another partition for running the security server.

The name of the Basic Security Manager (BSM) server routine is BSTPSTS. If an External Security Manager (ESM) also requires a server partition like the BSM, the name BSTPSTS must be replaced by the name of the ESM server routine in the server partition startup procedure $BJCLxxx, which is the default.

BSSINIT is the common security initialization routine for the BSM or an ESM. The parameter setting in the IPL SYS command controls the initialization process. If neither an ESM initialization phase (SYS ESM=phase) nor recovery mode (SYS SEC=RECOVER) has been specified, BSSINIT will start BSM initialization. BSSINIT initializes and starts partition FB as BSM server partition unless another partition than FB has been specified in the IPL SYS command (SYS SERVPART=xx).

If the IPL SYS command includes SEC=YES, a minimum protection of libraries and files is active during startup. This protection is based on DTSECTAB definitions and is in effect until the BSM or an ESM is active and takes over protection. The BSM requires the security server.

Return Codes from BSSINIT

The return codes of the common security initialization routine BSSINIT provide status information for conditional processing during startup in the $xJCLxxx procedure. Note that these return codes are no error indicators.

```
0         Do nothing for security (security may be inactive)
1 to 11   Identifies the server partition (11, for example, is the FB partition)
99        Indicates that the current partition is the server partition
```

Other Startup Programs

Besides startup program DTRISTRT, the following programs are also involved in startup processing. For tailoring startup, you should know what they are used for:

DTRIBASE
Identifies system characteristics such as teleprocessing access method and environment number.

DTRIINIT
Loads jobs into the VSE/POWER reader queue.

This program is a general utility program. The manual *z/VSE System Utilities* describes the program in detail under "DTRIINIT Utility".

DTRISCPU
Compares the CPU ID given as input parameter with the actual CPU ID. Called by $COMVAR if more than one CPU involved.

DTRSETP
Updates startup procedure CPUVARn.PROC.

IESWAITT
Waits an unlimited time until VTAM is active.

IESWAIT
Waits the specified number of seconds to allow the operator to request a startup mode. The number of seconds is passed as parameter with the PARM operand. See also parameter 6 under "Tracing Startup Processing" on page 26.

Tracing Startup Processing

z/VSE provides a parameter (TEST1) that allows you to trace the input processed and the output created by the startup program DTRISTRT. You can use this parameter to test your startup modifications. You call DTRISTRT with the following statement:

```
// EXEC DTRISTRT,SIZE=AUTO,PARM='CPUVAR2;;;;TEST1'
```

The parameters that can be specified are positional and must be enclosed in single quotes (′)and separated by semicolons (;). The following parameters (PARM=) are supported:

Parameter 1: **Name** of SETPARM procedure (CPUVAR2).

Parameter 2: Used by z/VSE for BASIC startup.

Parameter 3: Used by z/VSE for MINI startup.

Parameter 4: Reserved.

Parameter 5: **Trace** request (TEST1).

Parameter 6: Number of seconds (two digits) for operator to request startup mode (IESWAIT). The default is 10 seconds.

The parameters of interest here are 1 (CPUVAR2) and 5 (TEST1). Specifying TEST1 for parameter 5 causes the trace to be activated. The trace information is shown on SYSLOG and printed on SYSLIST. Figure 6 shows a sample portion of a startup trace.

```
IESI0216I LOG DTRISTRT USING MEMBER CPUVAR2.PROC IN IJSYSRS.SYSLIB.
IESI0231I SYNTAX ERROR IN STATEMENT "SETPARM ...  ". STATEMENT WILL BE IGNORED.
IESI0211I ALL PARTITIONS WILL BE INITIALIZED IN xxxxxx START MODE. IF YOU
          WANT TO INTERRUPT ENTER "MSG BG".
IESI0217I LOG DTRISTRT INPUT FROM MEMBER : XUSEBG =B0.
IESI0217I LOG DTRISTRT INPUT FROM MEMBER : XBASIC =NONE.
IESI0217I LOG DTRISTRT INPUT FROM MEMBER : XCOLD =NONE.
IESI0218I LOG DTRISTRT PROCESSING BG DECIDES ON RECOV STARTUP MODE.
IESI0219I LOG DTRISTRT OUTPUT INTO MEMBER : XMODEBG=RECOV.
IESI0219I LOG DTRISTRT OUTPUT INTO MEMBER : XPARTB0=BG.
IESI0217I LOG DTRISTRT INPUT FROM MEMBER : XUSEFB =NONE.
IESI0219I LOG DTRISTRT OUTPUT INTO MEMBER : XMODEFB DELETED.
IESI0217I LOG DTRISTRT INPUT FROM MEMBER : XUSEFA =NONE.
IESI0219I LOG DTRISTRT OUTPUT INTO MEMBER : XMODEFA DELETED.
   ⋮
```

Figure 6. Portion of a Startup Trace

DTRISTRT issues the following return codes:

00 Successful processing (not MINI startup).

01 Successful processing (MINI startup).

08 Function partially executed (not MINI startup). Processing continues. Possible errors:
  Maximum number of variables exceeded.
  Syntax error in SETPARM statement.

09 Function partially executed (MINI startup). Processing continues. For possible errors see return code 08.

12 Error occurred. Processing is terminated. Possible errors:
  Parameter syntax incorrect.

Library full on output.

16 Severe error occurred. Processing is terminated. Possible errors:
Phase not found.
GETVIS space exhausted.

Modifying Startup Processing Using CPUVARn Information

For example, you may want to process a user-written procedure in case of a specific partition startup condition. The following is assumed:

```
Partition: F8
CPU:       CPU1
Procedure: $8JCL.PROC
SETPARM procedure: CPUVAR1
Condition: COLD startup
```

To identify a COLD startup, you have to retrieve the corresponding system variable from CPUVAR1. Your statements in $8JCL may look as follows:

```
// EXEC PROC=CPUVAR1,XMODEF8
// IF &XMODEF8 = COLD THEN
// GO TO USRPROC
         .
         .
         .
/. USRPROC
// EXEC PROC=MYPROC
         .
         .
         .
```

Modifying Startup When Installing an Additional Program

Most likely you want to install additional programs on top of your z/VSE system. For example, z/VSE optional programs or your own application programs.

For startup, the following changes are required:

1. Update system variables XUSEyy and XSTATyy (set to INACTIVE) in CPUVARn.PROC for the partition used with the utility program DTRSETP. Refer to the manual *z/VSE System Utilities* under “DTRSETP Utility” for details about how to use this utility.
2. Catalog the startup job into a VSE sublibrary and load it into the VSE/POWER reader queue by using skeleton SKLOAD described under “Skeleton for Loading a Job (SKLOAD)” on page 62. Add the startup job also to the COLDJOBS load list for a COLD startup by using skeleton SKCOLD described under “Skeleton for Loading User Jobs During a COLD Startup (SKCOLD)” on page 61.
3. Update USERBG.PROC by including the name of the startup job by using skeleton SKUSERBG.
4. Update the LIBDEF chain if necessary.

Using Synchronization Points

Program DTRSETP provides a WAIT function that allows you to synchronize partitions at the JCL level. For example, partition F5 waits for partition F4 to reach a certain point in processing. To use this function, you must add a variable to the startup procedure CPUVARn. For example, USYNC01. Use program DTRSETP to add such a variable.

During system startup USYNC01 must be reset; to the value NO, for example. To do this, you must insert in the BG ASI procedure ($0JCL) an EXEC statement for program DTRSETP. Insert the statement after the JOB statement and before the EXEC DTRISTRT statement. Use skeleton SKJCL0 for that purpose.

The program running in F5 must use operation WAIT of program DTRSETP to initiate a wait loop for checking repeatedly the status of USYNC01. If the program running in F4 has reached its particular point of processing, it must use operation SET of program DTRSETP to set USYNC01 to an agreed value. This value will be recognized by the program running in F5 and synchronization can occur.

Refer to the manual *z/VSE System Utilities* under “DTRSETP Utility” for a detailed description of program DTRSETP.

Synchronizing Partition Startup Using IESWAITR Procedure

IESWAITR may be used to synchronize startup of TCP/IP and CICS, for example, if the CICS TS Web Support (CWS) is to be used. If IESWAITR is coded in the CICS startup (see also skeletons SKCICS and SKCICS2), it ensures that TCP/IP is up and running when CICS is started. IESWAITR can also be used for other components such as the DB2 Server.

Changing Startup for DASD Sharing

For an environment with DASD sharing (sharing any disk devices among two or more CPUs, for example SCSI disks), you must create procedures CPUVAR2 through CPUVARn by tailoring procedure $COMVAR accordingly. Skeleton SKCOMVAR can be used to tailor procedure $COMVAR. Refer to “Skeleton for Tailoring $COMVAR Procedure (SKCOMVAR)” on page 62 for details.

For general guidelines for DASD sharing, refer to the manual *z/VSE Guide to System Functions* under “DASD Sharing with Multiple VSE Systems”.

Changing Startup When Lock File Is Stored On SCSI DASD

Related Section:
- “Using Shared SCSI Disks” on page 91

In an environment with DASD sharing where the lock file is stored on a SCSI DASD (sharing disk devices among two or more CPUs), you must use the dialog *Tailor IPL Procedure* (Fastpath **243**) to define a *physical* FCP adapter for the lock file.

To define a lock file on a SCSI disk, you must:
1. In the *Tailor IPL Procedure* dialog, enter ’**2**’ (= ALTER) next to the IPL procedure you wish to change, and press Enter.
2. Enter a ’**1**’ next to DLF (“Modify Lock File Definition”) and press Enter. The *Lock File Definition* panel is then displayed:

```
 TAS$ICM4        TAILOR IPL PROCEDURE: LOCK FILE DEFINITION

 Enter the required data and press ENTER.

The  DLF  command  must be used if DASD sharing is specified in the supervisor.
If DASD sharing is not specified, any DLF command must be deleted.

VOLUME SERIAL................ WORK01      Volume Serial of lock file DASD
                                        (Enter blanks to delete command)
NUMBER OF CPUS.............. 9          Number of CPUs to be shared
START ADDRESS................ 200       Starting cylinder or block number
LOCK FILE SIZE............... 80        Number of cylinders or blocks (leave
                                        blank for VSE defaults)
SECURED DATA SET ?........... 1         2 = no, 1 = yes
FORMAT....................... 2         Formatting required (2 = no, 1
                                        = yes)
OWN FCP...................... ___       cuu of the FCP adapter,if lock file
                                        on SCSI
 WARNING: If the lock file was previously defined by  another  system,  specify
only the volume serial number and leave the other parameters blank.
 PF1=HELP      2=REDISPLAY  3=END
```

3. In the field “OWN FCP” you must enter the address of the FCP adapter you wish to use. You *must* specify the address of the FCP adapter used for connecting this SCSI DASD.

   Note: This operand will be ignored for non-SCSI disks (for example, ECKD or any other FBA device).
4. After you press Enter, the IPL procedure (for example, $IPLESA) will be updated with the above details. It is catalogued in the IJSYSRS library.

For further details about implementing SCSI support, refer to the chapter “Using SCSI Disks With Your z/VSE System” in the manual *z/VSE Planning*.

Using Skeletons for Tailoring System Startup

z/VSE provides skeletons that help you tailor startup procedures and jobs. The skeletons are provided in VSE/ICCF library 59 and reflect the original startup members as provided by z/VSE for the predefined environments.

Note: Before you change a skeleton, copy it to your primary VSE/ICCF library. Make your changes to that copy, not to the original. For details refer to “Copying Skeletons” on page 6.

On the following pages you find the statements of each startup skeleton listed. Those variables and names that have to be changed or that are likely candidates for changes are described and printed in bold. In general, you can change any statement of a skeleton (except for those mentioned under “Procedures and Jobs You Should Not Change” on page 17). You must then find out, however, how such changes impact the statements of other skeletons and change them accordingly. Following is a list of skeletons available for tailoring system startup.

- Skeletons for static partition allocations:
  - SKALLOCA = Environment A (entry system)
  - SKALLOCB = Environment B (medium system)
  - SKALLOCC = Environment C (large system)
- Skeletons for starting up partitions:
  - SKJCL0 = BG partition
  - SKJCL1 = F1 partition (VSE/POWER)
  - SKJCL2-SKJCL9 = F2-F9 partitions
  - SKJCLA = FA partition
  - SKJCLB = FB partition
  - SKJCLDYN = Dynamic partitions
- Skeletons for called procedures and jobs:
  - SKUSERBG = Job release and LIBDEF processing
  - SKPWSTRT = VSE/POWER autostart
  - SKLIBCHN = Define library search chains (LIBDEFs)
  - SKCICS = Startup job CICS Transaction Server and VSE/ICCF (F2)
  - SKVTAM = Startup job VTAM (F3)
  - SKTCPSTR = Startup job TCP/IP (F7)
  - SKCOLD = Adding jobs for COLD startup
  - SKLOAD = Load job into VSE/POWER reader queue
  - SKCOMVAR = DASD sharing
  - SKVTASTJ = Startup of the Virtual Tape Server partition
  - SKVCSSTJ = Startup of the Connector Server partition
- Skeletons for ***environments B and C***
  - SKCICS2 = Startup job second CICS Transaction Server (without VSE/ICCF). For details, see Chapter 8, “Installing a Second Predefined CICS Transaction Server,” on page 191.
  - SKCICSOL = Startup job CICS/VSE 2.3 (without VSE/ICCF). For details, see Chapter 9, “Creating a CICS Coexistence Environment,” on page 209.

For skeletons related to the z/VSE connectors support, refer to the *z/VSE e-business Connectors User’s Guide*.

Skeleton for Cataloging Startup Changes (SKENVSEL)

Figure 7 shows the statements of skeleton *SKENVSEL*. With it, you catalog changed startup procedures and jobs in library IJSYSRS.SYSLIB. In addition, copies of the cataloged startup members are saved in library PRD2.SAVE.

```
* $$ JOB JNM=ENVCAT,DISP=D,CLASS=0
// JOB ENVCAT
// EXEC LIBR,PARM='MSHP'
ACCESS S=IJSYSRS.SYSLIB
* $$ SLI ICCF=(SKALLOCA),LIB=(YY)
* $$ SLI ICCF=(SKALLOCB),LIB=(YY)
* $$ SLI ICCF=(SKALLOCC),LIB=(YY)
* $$ SLI ICCF=(SKJCL0),LIB=(YY)
* $$ SLI ICCF=(SKJCL1),LIB=(YY)
* $$ SLI ICCF=(SKJCL2),LIB=(YY)
* $$ SLI ICCF=(SKJCL3),LIB=(YY)
* $$ SLI ICCF=(SKJCL4),LIB=(YY)
* $$ SLI ICCF=(SKJCL5),LIB=(YY)
* $$ SLI ICCF=(SKJCL6),LIB=(YY)
* $$ SLI ICCF=(SKJCL7),LIB=(YY)
* $$ SLI ICCF=(SKJCL8),LIB=(YY)
* $$ SLI ICCF=(SKJCL9),LIB=(YY)
* $$ SLI ICCF=(SKJCLA),LIB=(YY)
* $$ SLI ICCF=(SKJCLB),LIB=(YY)
* $$ SLI ICCF=(SKBGSTRT),LIB=(YY)
* $$ SLI ICCF=(SKJCLBNT),LIB=(YY)
* $$ SLI ICCF=(SKJCLDYN),LIB=(YY)
* $$ SLI ICCF=(SKUSERBG),LIB=(YY)
* $$ SLI ICCF=(SKPWSTRT),LIB=(YY)
* $$ SLI ICCF=(SKLIBCHN),LIB=(YY)
/*
/&
* $$ EOJ
```

Figure 7. Skeleton SKENVSEL for Cataloging Startup Changes

Skeleton SKALLOCC is used with the large Environment C, that was introduced with z/VSE 3.1.

Note: Several skeletons for startup are not included in SKENVSEL for the following reasons:

- They contain the necessary JCL and JECL statements for cataloging.
- Their output is cataloged in IJSYSRS.SYSLIB and loaded directly into the VSE/POWER reader queue (skeletons SKCICS, SKCICS2, SKCICSOL, and SKVTAM).

Each *** $$ SLI** statement includes a startup procedure created by a skeleton. Delete all SLI statements for skeletons that you did not modify. For example, you must delete 2 of the 3 statements for partition allocation procedures. The **YY** variable indicates your VSE/ICCF primary library that contains your modified copy of the skeleton.

After you have modified the skeleton, enter the following command from the editor's command line:

```
@DTRSEXIT
```

This command calls a macro that deletes specific comments from the file. You should do this *before* filing the skeleton.

Finally, submit job ENVCAT for processing.

Skeletons for Static Partition Allocations

Figure 8 on page 33, Figure 9 on page 34, Figure 10 on page 35 show the statements of the skeletons provided for static partition allocations. The comments included in the skeletons are not shown.

Note on ALLOC R and SETPFIX

In releases prior to VSE/ESA 1.3, the skeletons for static partition allocations included ALLOC R definitions. Since VSE/ESA 1.3, the ALLOC R definition has been replaced by the JCL command SETPFIX. SETPFIX and ALLOC R are to be used as follows:

- SETPFIX command is to be used to set limits per partition for fixing pages (PFIX).
- An ALLOC R definition is to be used only to define storage per partition available for programs to be executed in **real mode**.

SETPFIX definitions are included in the startup procedures $1JCL (for VSE/POWER), VTAMSTRT (for VTAM), and CICSICCF (for CICS); the associated skeletons are SKJCL1, SKVTAM, and SKCICS. For TCP/IP the skeleton used is SKTCPSTR. For the second predefined CICS, the name of the startup procedure is CICS2, the name of the skeleton is SKCICS2. The name of the startup procedure for a coexisting CICS/VSE 2.3 is CICSOLD with the associated skeleton SKCICSOL.

Each skeleton's CATALOG statement shows the procedure name **ALLOC**. z/VSE uses this name for the allocation procedures provided, however, you may use your own name instead. To help you make a decision, read first "Considerations for Naming Conventions" on page 18. If you decide to use your own name, you must replace ALLOC (in skeletons SKALLOCx and SKJCL0) with the name you have chosen.

TCP/IP using OSAX-links require approximately 1 MB PFIX storage (above 16 MB) per link. For further details, refer to the skeleton SKTCPSTR.

After you have modified the skeleton, enter the following command from the editor's command line:

```
@DTRSEXIT
```

This command calls a macro that deletes specific comments from the skeleton. You should do this *before* filing the skeleton.

Use skeleton **SKENVSEL** for cataloging your changes. Refer to "Skeleton for Cataloging Startup Changes (SKENVSEL)" on page 31 for details.

Skeleton SKALLOCA

This skeleton applies to predefined environment A including 12 static partitions (one crossing the 16 MB line).

```
CATALOG ALLOC.PROC   DATA=YES       REPLACE=YES
ALLOC F1=4M
SIZE F1=1280K
ALLOC BG=6M
SIZE BG=1280K
ALLOC F2=30M
SIZE F2=2048K
ALLOC F3=15M
SIZE F3=600K
ALLOC F4=1M
SIZE F4=768K
ALLOC F5=1M
SIZE F5=768K
ALLOC F6=512K
SIZE F6=256K
ALLOC F7=16M
SIZE F7=1M
ALLOC F8=6M
SIZE F8=3584K
ALLOC F9=512K
SIZE F9=256K
ALLOC FA=512K
SIZE FA=256K
ALLOC FB=1M
SIZE FB=512K
SYSDEF DSPACE,DSIZE=15M
NPGR BG=100,F2=255,F3=100,F4=100,F5=50,F6=50,F7=50,F8=200
NPGR F9=50,FA=50,FB=50
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY ALLOC.PROC         REPLACE=YES
```

Figure 8. Skeleton SKALLOCA (Static Partition Allocations)

Note: The SYSDEF command defines the VTAM and VTAM application requirements for data spaces. This includes the VTAM requirements for VSE/POWER and CICS. This is also reflected in the // EXEC IPWPOWER,... statements of the VSE/POWER startup procedure (skeleton SKPWSTRT), the // EXEC DFHSIP,... statements of the CICS startup procedure (skeleton SKCICS), and the // EXEC ISTINCVT,... statement of the VTAM startup procedure (skeleton SKVTAM).

The value defined for DSIZE is a minimum value which must not be reduced. Add your own requirements for data spaces to this predefined value. The DSIZE value given here reflects the status at the time the manual was printed. The actual value shipped with z/VSE may be higher.

Skeleton SKALLOCB

This skeleton applies to predefined environment B prepared for a second CICS TS and a CICS coexistence environment. It includes 12 static partitions (four crossing the 16 MB line).

```
CATALOG ALLOC.PROC   DATA=YES       REPLACE=YES
ALLOC BG=6M
SIZE BG=1280K
ALLOC F1=5M
SIZE F1=1280K
ALLOC F2=50M
SIZE F2=2M
ALLOC F3=15M
SIZE F3=600K
ALLOC F4=20M
SIZE F4=2M
ALLOC F5=1M
SIZE F5=768K
ALLOC F6=512K
SIZE F6=256K
ALLOC F7=20M
SIZE F7=1M
ALLOC F8=50M
SIZE F8=2M
ALLOC F9=512K
SIZE F9=256K
ALLOC FA=512K
SIZE FA=256K
ALLOC FB=1M
SIZE FB=512K
SYSDEF DSPACE,DSIZE=20M
NPGR BG=100,F2=255,F3=100,F4=200,F5=50,F6=50,F7=100,F8=200
NPGR F9=50,FA=50,FB=50
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY ALLOC.PROC         REPLACE=YES
```

Figure 9. Skeleton SKALLOCB (Static Partition Allocations)

The SYSDEF command defines the VTAM and VTAM application requirements for data spaces. This includes the requirements for VSE/POWER and CICS. This is also reflected in the:

- // EXEC IPWPOWER,... statements of the VSE/POWER startup procedure (skeleton SKPWSTRT).
- // EXEC DFHSIP,... statements of the CICS startup procedures (skeletons SKCICS, SKCICS2, and SKCICSOL).
- // EXEC ISTINCVT,... statement of the VTAM startup procedure (skeleton SKVTAM).

The value defined for DSIZE is an average size covering most application needs. It leaves about 4 MB for non-VTAM applications like TCP/IP for VSE/ESA or CICS shared data tables. The actual value shipped with z/VSE may be higher.

Skeleton SKALLOCC

This skeleton applies to predefined environment C prepared for a large environment. It includes 12 static partitions (eleven crossing the 16 MB line).

```
CATALOG ALLOC.PROC   DATA=YES       REPLACE=YES
ALLOC BG=32M
SIZE BG=1280K
ALLOC F1=32M
SIZE F1=1280K
ALLOC F2=256M
SIZE F2=2M
ALLOC F3=15M
SIZE F3=600K
ALLOC F4=32M
SIZE F4=2M
ALLOC F5=32M
SIZE F5=1M
ALLOC F6=32M
SIZE F6=1M
ALLOC F7=32M
SIZE F7=1M
ALLOC F8=512M
SIZE F8=2M
ALLOC F9=32M
SIZE F9=1M
ALLOC FA=32M
SIZE FA=1M
ALLOC FB=2M
SIZE FB=512K
SYSDEF DSPACE,DSIZE=256M
NPGR BG=100,F2=255,F3=100,F4=200,F5=50,F6=50,F7=100,F8=200
NPGR F9=50,FA=50,FB=50
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY ALLOC.PROC         REPLACE=YES
```

Figure 10. Skeleton SKALLOCC (Static Partition Allocations)

The SYSDEF command defines the VTAM and VTAM application requirements for data spaces. This includes the requirements for VSE/POWER and CICS. This is also reflected in the:

- // EXEC IPWPOWER,... statements of the VSE/POWER startup procedure (skeleton SKPWSTRT).
- // EXEC DFHSIP,... statements of the CICS startup procedures (skeletons SKCICS, SKCICS2, and SKCICSOL).
- // EXEC ISTINCVT,... statement of the VTAM startup procedure (skeleton SKVTAM).

The value defined for DSIZE is an average size covering most application needs. It leaves about 240 MB for non-VTAM applications like TCP/IP for VSE/ESA or CICS shared data tables. The actual value shipped with z/VSE may be higher.

Skeletons for Starting Up BG Partition

You can use the skeletons *SKJCL0* and *SKUSERBG* to tailor startup for the BG partition. The comments included in the skeletons are not shown in the following figures. To understand the functions included in the startup for the BG partition, refer to "Overview of Startup Processing" on page 12.

Both skeletons are shipped in VSE/ICCF library 59. Figure 11 shows the contents of skeleton SKJCL0 and Figure 12 on page 42 the contents of skeletons SKUSERBG. Certain AR commands can be included or are recommended for inclusion in the $0JCL procedure. They are listed under "Including AR Commands" on page 41.

Skeleton SKJCL0

$0JCL is the procedure name used by z/VSE but you may use your own name instead. To help you make a decision, read first "Considerations for Naming Conventions" on page 18.

Note: The name for this procedure must always start with **$0** and the following six characters must be equal for all related JCL startup procedures.

If you decide to use your own name, you must update $ASIPROC (if used) or the operator must interrupt IPL processing to enter the correct name.

```
CATALOG $0JCL.PROC   DATA=YES   REPLACE=YES
STDOPT ACANCEL=NO,DECK=NO,DUMP=PART,SYSDUMP=YES,SXREF=YES
SYSDEF DSPACE,DSIZE=15M
// EXEC PROC=STDLABEL                   CALLS ALSO STDLABUP AND STDLABUS LOAD DISK
// VDISK UNIT=FDF,BLKS=2880,VOLID=VDIDLA,USAGE=DLA
*  VDISK UNIT=CUU,BLKS=81920,VOLID=VDIWRK
// EXEC PROC=STDLABEL                   CALLS ALSO STDLABUP AND STDLABUS LOAD VDISK
// EXEC PROC=SETSDL                     SET SDL
PRTY BG,FA,F9,F8,F6,F5,F4,F2,F7,FB,F3,F1
ASSGN SYSLST,IGN
// JOB BGINIT
// SETPARM XNCPU=' '
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRISTRT,SIZE=AUTO,PARM='CPUVAR&XNCPU;$$JCLBSX;$$JCLMIN'
/*
// SETPARM RETCODE=$RC
// SETPARM XSPINIT='FINISHED '
// SETPARM XMODEBG='MINI'
// SETPARM XPARTPW='F1'
// SETPARM XPWMODE='WARM'
// IF RETCODE=1 OR RETCODE=9 THEN
// GOTO ALLOCBSX
// EXEC PROC=CPUVAR&XNCPU,XMODEBG,XPARTPW,XPWMODE,XSPINIT
// IF XSPINIT ¬= FINISHED THEN
// GOTO NOSDL
```

Workfiles on Virtual Disk:

If you want to place workfiles on a virtual disk, you must activate the following statements in the skeleton:

```
*  VDISK UNIT=CUU,BLKS=81920,VOLID=VDIWRK
*  EXEC PROC=IESWORK      DEFINE WORK FILES ON VIRTUAL DISK   (on next page)
```

Figure 11. Skeleton SKJCL0 (Startup Procedure for BG Partition) (Part 1 of 5)

```
// EXEC PROC=LIBSDL                  PROVIDE CORRECT LIBDEF FOR SET SDL
SET SDL
LIST=$SVAVTAM
LIST=$SVACICS
LIST=$SVAREXX
LIST=$SVAASMA
LIST=$SVACONN
LIST=$SVACEE
LIST=$SVAEDCM
/*
* -------------------------------------------------------------------
* DEPENDING ON THE SVA SIZE AND ON THE LANGUAGES USED IN LE/VSE
* YOU MAY ADDITIONALLY LOAD RUNTIME SUPPORT FOR PL/I AND/OR COBOL.
* MOVE THE APPROPRIATE LIST STATEMENT ABOVE.
* ADDITIONAL AMOUNT OF STORAGE AS OF GA TIME:
*
*                                  R-MODE 24   RMODE=ANY
* LIST=$SVAIGZM    LE/VSE COBOL      0KB       129KB
* LIST=$SVAIBMM    LE/VSE PL/I      37KB       209KB
* * -------------------------------------------------------------------
// LIBDROP PHASE
/. NOSDL
EXPLAIN ON
// IF XMODEBG=BASIC THEN
// GOTO ALLOCBSX
// EXEC PROC=ALLOC     CHANGE TO PROCNAME DEFINED IN SKALLOCx  x=A,B,C
*  EXEC PROC=IESWORK      DEFINE WORK FILES ON VIRTUAL DISK
// GOTO SECSTRT
/. ALLOCBSX
// EXEC PROC=ALLOCBSX     ALLOCs FOR BASIC START
// SETPARM XPARTPW=F1
*         IF YOU CHANGED THE PRIORITY IN  YOUR NORMAL SYSTEM
*         YOU HAVE TO RESET IT FOR BASIC START TO:
*         PRTY BG,FA,F9,F8,F6,F5,F4,F2,F7,FB,F3,F1
EXPLAIN ON
// PAUSE
/. SECSTRT
// ON $RC=99 GOTO ERROR
// EXEC BSSINIT
/*
// SETPARM RETCODE=$RC
// IF RETCODE¬=0 THEN
// GOTO PARTFB
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=RECOVERY
/*
```

Figure 11. Skeleton SKJCL0 (Startup Procedure for BG Partition) (Part 2 of 5)

```
// GOTO PWRSTRT
/. PARTFB
// IF RETCODE¬=11 THEN
// GOTO PARTFA
// IF XPARTPW = FB THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=FB
/*
// GOTO PWRSTRT
/. PARTFA
// IF RETCODE¬=10 THEN
// GOTO PARTF9
// IF XPARTPW = FA THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=FA
/*
// GOTO PWRSTRT
/. PARTF9
// IF RETCODE¬=9 THEN
// GOTO PARTF8
// IF XPARTPW = F9 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F9
/*
// GOTO PWRSTRT
/. PARTF8
// IF RETCODE¬=8 THEN
// GOTO PARTF7
// IF XPARTPW = F8 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F8
../*
// GOTO PWRSTRT
/. PARTF7
// IF RETCODE¬=7 THEN
// GOTO PARTF6
// IF XPARTPW = F7 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F7
/*
// GOTO PWRSTRT
/. PARTF6
// IF RETCODE¬=6 THEN
// GOTO PARTF5
// IF XPARTPW = F6 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F6
/*
```

Figure 11. Skeleton SKJCL0 (Startup Procedure for BG Partition) (Part 3 of 5)

```
// GOTO PWRSTRT
/. PARTF5
// IF RETCODE¬=5 THEN
// GOTO PARTF4
// IF XPARTPW = F5 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F5
/*
// GOTO PWRSTRT
/. PARTF4
// IF RETCODE¬=4 THEN
// GOTO PARTF3
// IF XPARTPW = F4 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F4
/*
// GOTO PWRSTRT
/. PARTF3
// IF RETCODE¬=3 THEN
// GOTO PARTF2
// IF XPARTPW = F3 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F3
/*
// GOTO PWRSTRT
/. PARTF2
// IF RETCODE¬=2 THEN
// GOTO PARTF1
// IF XPARTPW = F2 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F2
/*
/. PARTF1
// IF RETCODE¬=1 THEN
// GOTO ERROR
// IF XPARTPW = F1 THEN
// GOTO ERROR
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XSECP=F1
/*
```

Figure 11. Skeleton SKJCL0 (Startup Procedure for BG Partition) (Part 4 of 5)

```
/. PWRSTRT
SET MRCZERO
START &XPARTPW
STOP
ASSGN SYSIN,FEC,PERM
ASSGN SYSPCH,FED
ASSGN SYSLST,FEE
ASSGN SYSLNK,DISK,VOL=DOSRES,SHR           SYSTEM LINK FILE
ASSGN SYS001,DISK,VOL=SYSWK1,SHR           SYSTEM WORK FILE 1
ASSGN SYS002,DISK,VOL=SYSWK1,SHR           SYSTEM WORK FILE 2
ASSGN SYS003,DISK,VOL=SYSWK1,SHR           SYSTEM WORK FILE 3
ASSGN SYS004,DISK,VOL=SYSWK1,SHR           SYSTEM WORK FILE 4
// IF XSPINIT ¬= FIRST THEN
// GOTO SKIP
* ==================================== *
*                                      *
*            INSTALLATION OF           *
// EXEC PROC=SPLEVEL
*                                      *
* ==================================== *
// EXEC PROC=LOADINST
// EXEC DTRSETP,PARM='CPUVAR1;;'
   SET XSPINIT=INSTALL
/*
// PWR PRELEASE RDR,INSTALL
// GOTO EXIT
/. SKIP
// IF XPWMODE=COLD OR XPWMODE=BASIC THEN
// GOTO COLDPART
// GOTO ENDCOLD
/. COLDPART
// ID USER=FORSEC                 !! NO PWD REQUIRED !!
// EXEC PROC=COLDJOBS
/. ENDCOLD
// IF XMODEBG ¬= MINI THEN
// GOTO NOTMINI
// EXEC PROC=MINIBG
// GOTO EXIT
/. NOTMINI
// IF XMODEBG ¬= BASIC THEN
// GOTO USER
// ID USER=FORSEC                !! NO PWD REQUIRED !!
// EXEC PROC=BASICBG
// GOTO EXIT
/. USER
// EXEC PROC=USERBG     CHANGE TO YOUR PROCNAME AS USED IN SKUSERBG
// GOTO EXIT
/. ERROR
* ERROR IN THE PARTITION SETUP FOR SECURITY SERVER. SERVER PARTITION
* MAY NOT BE &XPARTPW
/. EXIT
/&
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY $0JCL.PROC      REPLACE=YES
```

Figure 11. Skeleton SKJCL0 (Startup Procedure for BG Partition) (Part 5 of 5)

In a system with security (access control) active, the // ID statements for user FORSEC ensure that startup procedures COLDJOBS and BASICBG have the appropriate access rights. In an ASI procedure no password is required for the ID statement.

Refer to chapter Chapter 7, “Protecting Resources,” on page 129 for details about the z/VSE access control support.

If you modified procedure USERBG and specified a different procedure name, you must change the procedure name here as well. Use the name entered in the CATALOG statement of skeleton SKUSERBG. Use skeleton **SKENVSEL** for cataloging your changes. Refer to “Skeleton for Cataloging Startup Changes (SKENVSEL)” on page 31 for details.

Including AR Commands

AR (attention routine) commands such as the following can be included in the $0JCL startup procedure for the BG partition:

```
BANDID
CACHE
DEBUG
EXPLAIN
FREE
LFCB
LUCB
OFFLINE
ONLINE
OPERATE
PRTYIO
RESERV
SETDF
SYSECHO
TPBAL
```

You find a detailed description of these commands in the manual *z/VSE System Control Statements*.

Skeleton SKUSERBG

```
CATALOG USERBG.PROC DATA=YES REPLACE=YES
* START MODE FOR BG-PARTITION IS NORMAL
* ***************************************
*             YOUR SYSTEM IS            *
// EXEC PROC=SPLEVEL
* ***************************************
STDOPT DATE=MDY                            CHANGE STANDARD OPTIONS IF WANTED
// EXEC PROC=LIBDEF                        CHANGE TO YOUR OWN LIBDEF PROC
// EXEC IESIRCVT                           INIT RCVT, REMOVE IN CASE OF AN ESM
/*
// EXEC ARXLINK                            INITIALIZE REXX/VSE
// LIBDEF DUMP,CATALOG=SYSDUMP.BG,PERM
*  TO ENABLE DB2, ENTER KEY AND CUSTOMER INFO, REMOVE ASTERISKS TO
*  ACTIVATE, CONTINUATION LINE START COLUMN 16.
*  EXEC IVALPKEY,PARM='PRODUCT=DB2 KEY=0000-1111-2222-3333-4444 CUSTINF*
*             O=C111-111-1111'
/*
// SETPARM XNCPU=''
// EXEC PROC=$COMVAR,XNCPU                 GET CPU NUMBER
// SETPARM XENVNR=''
// EXEC PROC=CPUVAR&XNCPU,XENVNR
// PWR PRELEASE RDR,VTAMSTRT       OR YOUR VTAM (SKVTAM)
// EXEC IESWAIT,PARM='03'
/*
// IF SSLCAUT NE YES THEN
// GOTO NOCAUT
// EXEC BSSDCERT,PARM='ACT'
/*
/. NOCAUT
// PWR PRELEASE RDR,CICSICCF       OR YOUR CICS (SKCICS)
// PWR PRELEASE RDR,CEEWARC        LE - AR INTERFACE
* // PWR PRELEASE RDR,CICS2        OR YOUR CICS2 SKCICS2)
* // PWR PRELEASE RDR,CICSOLD      OR YOUR CICSOLD (SKCICSOL)
* // PWR PRELEASE RDR,TCPIP00      OR YOUR TCP/IP STARTUP
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY USERBG.PROC REPLACE=YES
```

Figure 12. Skeleton SKUSERBG (Startup Procedure for BG Partition)

USERBG is the procedure name used by z/VSE. If you modify the procedure, you may want to change the procedure name as well. To help you make a decision, read first "Considerations for Naming Conventions" on page 18.

If you decide to use your own name, you must also update procedure $0JCL. Use skeleton SKJCL0 and modify the corresponding EXEC PROC statement.

Procedure USERBG calls procedure LIBDEF and releases the startup jobs VTAMSTRT and CICSICCF (and CICS2 if installed). If you modified any of these and specified your own names, replace these names accordingly:
- LIBDEF
- VTAMSTRT
- CICSICCF
- CEEWARC
- CICS2
- CICSOLD

Use the name you entered in the CATALOG statement of the procedure or job changed. The names of the skeletons are:

- SKLIBCHN
- SKVTAM
- SKCICS
- SKCICS2
- SKCICSOL
- SKTCPSTR.

Use skeleton **SKENVSEL** for cataloging your changes. Refer to "Skeleton for Cataloging Startup Changes (SKENVSEL)" on page 31 for details.

Enabling the DB2 Server for VSE

To enable the DB2 Server for VSE you must activate the EXEC IVALPKEY statement. Refer also to "System Provided Application Profiles" on page 117.

Skeletons for Starting Up VSE/POWER

You can use skeletons *SKJCL1* and *SKPWSTRT* to modify the startup of VSE/POWER in partition F1.

Both skeletons are shipped in VSE/ICCF library 59. Figure 13 and Figure 14 on page 47 show the contents of the skeletons. Comments included in the skeletons are not shown.

Skeleton SKJCL1

```
CATALOG $1JCL.PROC DATA=YES REPL=YES
// JOB POWSTART
// OPTION SADUMP=5
// EXEC PROC=DTRICCF                    ASSIGNMENTS FOR DTSFILE
// EXEC PROC=DTRPOWR                    ASSIGNMENTS FOR VSE/POWER
// SETPARM DASD=''
// SETPARM XNCPU=''
// SETPARM XENVNR=''
// SETPARM XPWMODE=''
// SETPARM XPARTPW=''
// SETPARM XSECP=''
// SETPARM XSPINIT=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC PROC=CPUVAR&XNCPU,DASD,XENVNR,XPWMODE,XPARTPW,XSECP,XSPINIT
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTAT&XPARTPW=ACTIVE'
/*
// EXEC PROC=LIBDEF
// EXEC PROC=LIBDUMP
// SETPFIX LIMIT=200K
// ID USER=FORSEC                       !! NO PWD REQUIRED !!
// EXEC PROC=POWSTRT&XENVNR,XPWMODE,XSECP
// ID USER=DUMMY                        !! NO PWD REQUIRED !!
/*
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTAT&XPARTPW=INACTIVE'
/*
// IF XSPINIT = FIRST OR XSPINIT = INSTALL THEN
// GOTO NOSEC
* -------------------------------------------------------------
* SECURITY SERVER PARTITION WILL BE STOPPED
* YOU MAY ENTER A PAUSE STATEMENT HERE IN CASE YOU DON'T WANT
* TO ALWAYS STOP. YOU WOULD HAVE TO ENTER // GOTO NOSEC IN
* CASE YOU DON'T WANT TO STOP.
* -------------------------------------------------------------
// EXEC DTRIATTN,PARM='MSG &XSECP,DATA=STOPNOREP'
/*
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;SET XSECP=RECOVER'
/*
/. NOSEC
/&
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY $1JCL.PROC         REPLACE=YES
```

Figure 13. Skeleton SKJCL1 (Startup Procedure for VSE/POWER Partition)

If you modify this procedure, you may also change the procedure name $1JCL. To help you make a decision, read first “Considerations for Naming Conventions” on page 18. Note that the name for this procedure **must** always start with **$1** and the following six characters must be equal for all related JCL startup procedures.

In a system with security (access control) active, the // ID statement for user FORSEC ensures the appropriate access rights for procedure POWSTRTn. The //

ID statement for user DUMMY turns off these access rights. In an ASI procedure no password is required for the ID statement.

Refer to Chapter 7, “Protecting Resources,” on page 129 for details about the z/VSE access control support.

If you decide to use your own name, you must update $ASIPROC (if used) or the operator must interrupt IPL processing to enter the correct name.

For an explanation of the SETPFIX definition, refer to “Skeletons for Static Partition Allocations” on page 32. The EXEC PROC statement for LIBDEF calls the procedure that defines library search chains and assignments for the partitions which are controlled by VSE/POWER. If you modified the LIBDEF procedure (skeleton SKLIBCHN) and specified your own procedure name, change this statement accordingly. Use the name you entered in the CATALOG statement of SKLIBCHN.

The EXEC PROC statement for POWSTRT calls the procedure that defines warm and cold starts. &XENVNR identifies the predefined environment chosen (A, B, or C). If you modified this procedure (skeleton SKPWSTRT) and specified your own procedure name, change this statement accordingly. Use the name you entered in the CATALOG statement of SKPWSTRT.

After you have modified the skeleton, enter the following command from the editor’s command line to delete specific comments from the skeleton:

```
@DTRSEXIT
```

Use skeleton *SKENVSEL* for cataloging your changes. Refer to “Skeleton for Cataloging Startup Changes (SKENVSEL)” on page 31 for details.

Skeleton SKPWSTRT

This skeleton is in two parts; one for a COLD, the other for a WARM startup. The n in POWSTRTn.PROC reflects the predefined environment selected during initial installation: A, B, or C.

If you modify this procedure, you may also change the procedure name POWSTRTn (n identifies the environment selected during initial installation). Refer also to “Considerations for Naming Conventions” on page 18. If you decide to use your own name, you must update the corresponding EXEC PROC statement in procedure $1JCL (skeleton SKJCL1).

The statements

```
SET SYSID=Y
SET PNET=YYYYYYYY
```

for shared spooling and networking are optional. Delete them if you do not use these functions.

The statement

```
SET SECNODE=AAAA
```

is for a multiple-node environment with security (access control) active. It is required to distinguish the nodes for security processing.

The statement

```
SET SECAC=NO
```

indicates that the job and its output are not Spool Access Protected. NO is the default value.

The statement:

```
SET SJECL=YES
```

supports the cataloging of VSE/POWER jobs into a VSE library.

The statement:

```
SET WORKUNIT=PA
```

allows VSE/POWER to run on multiple CPUs in parallel.

Do not change the following statement:

```
DEFINE L,CICSDATA,3F00,1,255,*
```

It specifies for the CICS Report Controller a Resource Security Level (RSL) value for reports generated by batch programs. Refer to "DEFINE: Specifying User-Defined Output Operands"in the manual *VSE/POWER Administration and Operation* for a detailed description of the DEFINE command.

The other DEFINE statements are for PSF/VSE (Print Services Facility/VSE), a z/VSE optional program which provides advanced printing support for certain types of IBM printers.

The VSE/POWER

```
PLOAD DYNC,ID=n,FORCE
```

command (shown on the following page) is optional and loads the dynamic class table DTR$DYNn (DTR$DYNC is the name of the table shipped with z/VSE) defining the parameters required for activating dynamic partitions. Dynamic partitions are supported for all predefined environments.

Note: This skeleton assumes that the Security Server runs in partition FB. If you have selected another partition for the Security Server, which is not recommended, you need to modify the skeleton accordingly. If there is a need to select another partition, you must select a static partition which is not controlled by VSE/POWER.

```
CATALOG POWSTRTn.PROC    REPL=YES  DATA=YES
// EXEC DTRWAITP
// IF XPWMODE ¬= COLD THEN
// GOTO WARM
// IF XSECP = FB THEN
// GOTO COLDFB
// ASSGN SYSLST,UA
// EXEC IPWPOWER,DSPACE=2M
SET SYSID=Y
SET PNET=YYYYYYYY
SET SECNODE=AAAA
SET SECAC=NO
SET SJECL=YES
SET WORKUNIT=PA
DEFINE L,CICSDATA,3F00,1,255,*
DEFINE L,CKPTPAGE,4,1,2,B,1,32767
DEFINE L,FORMDEF,1D,1,6,C
DEFINE L,PAGEDEF,1F,1,6,C
DEFINE L,PIMSG,21,2,3,C
DEFINE L,DATACK,2022,1,8,C
DEFINE L,FORMS,10,1,4,C
DEFINE L,PRMODE,18,1,8,C
DEFINE L,TRC,1A,1,3,C
FORMAT=D,A
PSTART BG,A0I
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F2,L2
READER=FEC
PRINTERS=FEE,FEF
PUNCHES=FED
PSTART F3,K3
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F4,J4
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F5,H5
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F6,M6
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F7,N7
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F8,P8
READER=FEC
```

Figure 14. Skeleton SKPWSTRT (VSE/POWER Warm and Cold Starts) (Part 1 of 5)

```
PRINTERS=FEE
PUNCHES=FED
PSTART F9,Q9
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART FA,TV
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART FB,BU
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART DEV,YYYYYYYY,SYSPSFXX,P
PSTART TASKTR,ENAB,10
* PLOAD DYNC,ID=n,FORCE
/*
// GOTO EXIT
/. COLDFB            COLD START, SECURITY SERVER IS FB
// ASSGN SYSLST,UA
// EXEC IPWPOWER,DSPACE=2M
SET SYSID=Y
SET PNET=YYYYYYYY
SET SECNODE=AAAA
SET SECAC=NO
SET SJECL=YES
SET WORKUNIT=PA
DEFINE L,CICSDATA,3F00,1,255,*
DEFINE L,CKPTPAGE,4,1,2,B,1,32767
DEFINE L,FORMDEF,1D,1,6,C
DEFINE L,PAGEDEF,1F,1,6,C
DEFINE L,PIMSG,21,2,3,C
DEFINE L,DATACK,2022,1,8,C
DEFINE L,FORMS,10,1,4,C
DEFINE L,PRMODE,18,1,8,C
DEFINE L,TRC,1A,1,3,C
FORMAT=D,A
PSTART BG,A0I
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F2,L2
READER=FEC
PRINTERS=FEE,FEF
PUNCHES=FED
PSTART F3,K3
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F4,J4
READER=FEC
PRINTERS=FEE
PUNCHES=FED
```

Figure 14. Skeleton SKPWSTRT (VSE/POWER Warm and Cold Starts) (Part 2 of 5)

```
PSTART F5,H5
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F6,M6
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F7,N7
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F8,P8
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F9,Q9
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART FA,TV
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART DEV,YYYYYYYY,SYSPSFXX,P
PSTART TASKTR,ENAB,10
* PLOAD DYNC,ID=n,FORCE
/*
// GOTO EXIT
/. WARM              POWER WARM START
// IF XSECP = FB THEN
// GOTO WARMFB
// EXEC IPWPOWER,DSPACE=2M
SET SYSID=Y
SET PNET=YYYYYYYY
SET SECNODE=AAAA
SET SECAC=NO
SET SJECL=YES
SET WORKUNIT=PA
DEFINE L,CICSDATA,3F00,1,255,*
DEFINE L,CKPTPAGE,4,1,2,B,1,32767
DEFINE L,FORMDEF,1D,1,6,C
DEFINE L,PAGEDEF,1F,1,6,C
DEFINE L,PIMSG,21,2,3,C
DEFINE L,DATACK,2022,1,8,C
DEFINE L,FORMS,10,1,4,C
DEFINE L,PRMODE,18,1,8,C
DEFINE L,TRC,1A,1,3,C
FORMAT=NO
PSTART BG,A0I
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F2,L2
READER=FEC
```

Figure 14. Skeleton SKPWSTRT (VSE/POWER Warm and Cold Starts) (Part 3 of 5)

```
PRINTERS=FEE,FEF
PUNCHES=FED
PSTART F3,K3
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F4,J4
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F5,H5
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F6,M6
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F7,N7
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F8,P8
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F9,Q9
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART FA,TV
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART FB,BU
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART DEV,YYYYYYYY,SYSPSFXX,P
PSTART TASKTR,ENAB,10
* PLOAD DYNC,ID=N,FORCE
/*
// GOTO EXIT
/. WARMFB            WARM START, SECURITY SERVER IS FB
// EXEC IPWPOWER,DSPACE=2M
SET SYSID=Y
SET PNET=YYYYYYYY
SET SECNODE=AAAA
SET SECAC=NO
SET SJECL=YES
SET WORKUNIT=PA
DEFINE L,CICSDATA,3F00,1,255,*
DEFINE L,CKPTPAGE,4,1,2,B,1,32767
DEFINE L,FORMDEF,1D,1,6,C
DEFINE L,PAGEDEF,1F,1,6,C
DEFINE L,PIMSG,21,2,3,C
```

Figure 14. Skeleton SKPWSTRT (VSE/POWER Warm and Cold Starts) (Part 4 of 5)

```
DEFINE L,DATACK,2022,1,8,C
DEFINE L,FORMS,10,1,4,C
DEFINE L,PRMODE,18,1,8,C
DEFINE L,TRC,1A,1,3,C
FORMAT=NO
PSTART BG,A0I
READER=FEC
RINTERS=FEE
PUNCHES=FED
PSTART F2,L2
READER=FEC
PRINTERS=FEE,FEF
PUNCHES=FED
PSTART F3,K3
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F4,J4
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F5,H5
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F6,M6
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F7,N7
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F8,P8
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F9,Q9
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART FA,TV
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART DEV,YYYYYYYY,SYSPSFXX,P
PSTART TASKTR,ENAB,10
* PLOAD DYNC,ID=N,FORCE
/*
/. EXIT
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY POWSTRTn.PROC      REPLACE=YES
```

Figure 14. Skeleton SKPWSTRT (VSE/POWER Warm and Cold Starts) (Part 5 of 5)

There are four commands for each of the twelve partitions named in the skeleton. For example:

```
PSTART BG,A0I
READER=FEC
PRINTERS=FEE
PUNCHES=FED
```

These four commands are required for each partition. They define some of the partition's operating characteristics and its device configuration for spooling. The manual *VSE/POWER Administration and Operation* describes these commands in detail.

z/VSE uses the first PRINTERS/PUNCHES as default device. For that reason, your real device should precede FEE or FED, respectively.

If more than one device is specified for PRINTERS or PUNCHES, it is recommended to code the POWER * $$ LST and the * $$ PUN statements for device selection. Otherwise, the LST and PUN parameters are associated with the first device and the system will take VSE/POWER default values when spooling to a device other than the first device.

Be sure to delete the commands for the partition(s) that you do not want to have autostarted. Do this for both the cold and the warm start definitions.

You can use any device address. However, the device must be defined in the hardware configuration table. This is done either during initial installation by device sensing or by using the *Configure Hardware* dialog.

After you have modified the skeleton, enter the following command from the editor's command line:

```
@DTRSEXIT
```

This command calls a macro that deletes specific comments from the skeleton. You should do this *before* filing the skeleton.

Use skeleton **SKENVSEL** for cataloging your changes. Refer to "Skeleton for Cataloging Startup Changes (SKENVSEL)" on page 31 for details.

Skeleton for Defining Library Search Chains (SKLIBCHN)

You can use the skeleton *SKLIBCHN* to define search chains and assignments for the batch partitions controlled by VSE/POWER. Figure 15 on page 53 shows the contents of the skeleton. Comments included in the skeleton are not shown. The name of the procedure created by this skeleton is **LIBDEF**.

```
CATALOG   LIBDEF.PROC  DATA=YES  REPLACE=YES
// LIBDEF  PHASE,SEARCH=(PRD2.CONFIG,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD1.BASE,PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE,
               PRD2.COMM,PRD2.COMM2,PRD2.AFP,PRIMARY.$$C),
               CATALOG=YYYYYYY.YYYYYYYY,
               PERM
// LIBDEF    OBJ,SEARCH=(PRD2.CONFIG,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD1.BASE,PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE,
               PRD2.COMM,PRD2.COMM2,PRD2.AFP,PRIMARY.$$C),PERM
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD1.BASE,PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE,PRD2.COMM,
               PRD2.COMM2,PRD2.AFP,PRIMARY.$$C,PRD1.MACLIB),PERM
/*
/+
CATALOG   LIBDEFS.PROC  DATA=YES  REPLACE=YES
// LIBDEF  PHASE,SEARCH=(PRD2.CONFIG,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD1.BASED,PRD1.BASE,PRD2.SCEEBASD,PRD2.SCEEBASE,
               PRD2.PROD,PRD2.DBASE,PRD2.COMM,PRD2.COMM2,PRD2.AFP,
               PRIMARY.$$C),
               CATALOG=YYYYYYY.YYYYYYYY,
               PERM
// LIBDEF    OBJ,SEARCH=(PRD2.CONFIG,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD1.BASED,PRD1.BASE,PRD2.SCEEBASD,PRD2.SCEEBASE,
               PRD2.PROD,PRD2.DBASE,PRD2.COMM,PRD2.COMM2,PRD2.AFP,
               PRIMARY.$$C),PERM
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD1.BASED,PRD1.BASE,PRD2.SCEEBASD,PRD2.SCEEBASE,
               PRD2.PROD,PRD2.DBASE,PRD2.COMM,PRD2.COMM2,PRD2.AFP,
               PRIMARY.$$C,PRD1.MACLIBD,PRD1.MACLIB),PERM
/*
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY LIBDEF.PROC   REPLACE=YES
COPY LIBDEFS.PROC  REPLACE=YES
```

Figure 15. Skeleton SKLIBCHN (Defining Library Search Chains)

Note: The second part of skeleton SKLIBCHN, called LIBDEFS, includes in addition libraries PRD1.BASED, PRD2.SCEEBASD, and PRD1.MACLIBD. These libraries are required only when applying service (PTFs) indirectly. Further details about these libraries are provided in the manual *z/VSE System Upgrade and Service* under “Job Sequence for PTF Application”.

If you replace the procedure name LIBDEF by a name of your own, you must also change the name in skeletons such as SKJCL1, SKUSERBG, and SKINITNN.

Replace the **Y** strings in the three LIBDEF statements to define your permanent sublibrary search chains for phase, object, and source members. Note that you should also include the system sublibraries as defined in the skeleton. If you do not, the Interactive Interface may not function correctly. In the LIBDEF statement for member type PHASE, the **Y** string of the CATALOG operand defines the sublibrary that the linkage editor uses to catalog phases.

The skeleton reflects the search chains established after initial installation. If you do not install optional programs or if you install them in your own user libraries, some of the sublibraries remain empty and may be deleted. After initial installation the following sublibraries are empty:
- PRD2.COMM
- PRD2.COMM2
- PRD2.DBASE
- PRD2.PROD
- PRD2.AFP
- PRD2.DB2740
- PRD2.DFHDOC
- PRD2.DB2STP
- PRIMARY.$$C
- PRIMARY.SUF

The sublibrary PRD2.AFP is for the z/VSE optional program PSF/VSE and the related z/VSE optional programs OGL/370 and PPFA/370.

After you have modified skeleton SKLIBCHN, enter the following command from the editor’s command line:

```
@DTRSEXIT
```

This command calls a macro that deletes specific comments from the file. You should do this *before* filing the skeleton. Use skeleton *SKENVSEL* for cataloging your changes. Refer to “Skeleton for Cataloging Startup Changes (SKENVSEL)” on page 31 for details.

Skeleton for Starting Up the CICS Transaction Server and VSE/ICCF (SKCICS)

Skeleton *SKCICS* creates a startup job for the CICS Transaction Server and VSE/ICCF in default partition F2. If you want to create your own startup procedure for these components, use skeleton SKCICS. The job stream created with SKCICS catalogs the startup job CICSICCF into IJSYSRS.SYSLIB and loads it directly into the VSE/POWER reader queue.

If you have to perform a BASIC startup (because of CICS startup problems, for example) perform the following steps:
- Perform BASIC startup.
- Use skeleton SKCICS to create and catalog members CICSICCF.Z and LDCICS.PROC into IJSYSRS.SYSLIB.
- Perform a normal startup.
- Run procedure LDCICS.PROC to load the startup job CICSICCF into the VSE/POWER reader queue.
- Release startup job CICSICCF to continue normal startup.

```
* $$ JOB JNM=CATCICS,DISP=D,CLASS=0
// JOB CATCICS     CATALOG CICSICCF AND LDCICS, LOAD CICSICCF
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG  CICSICCF.Z        REPLACE=YES
$$$$ JOB JNM=CICSICCF,DISP=L,CLASS=2,EOJMSG=YES
$$$$ LST CLASS=A,DISP=D,RBS=100
// JOB CICSICCF         CICS/ICCF STARTUP
// OPTION SADUMP=5
// OPTION SYSDUMPC
// UPSI 11100000
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASED,PRD1.BASE,PRD2.PROD,        X
               PRD2.SCEEBASD,PRD2.SCEEBASE,PRD2.DBASE,PRD1.MACLIBD,    X
               PRD1.MACLIB),PERM
// LIBDEF DUMP,CATALOG=SYSDUMP.F2
// SETPARM XNCPU=''
// SETPARM XMODEF2=AUTO
// SETPARM XAPPLF2=''
// SETPARM XSPINIT=''
// SETPARM XENVNR=''
// SETPARM XSECP=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF2=ACTIVE'
$$/*
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XAPPLF2=DBDCCICS'
$$/*
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XPARTCI=F2'
$$/*
// EXEC PROC=CPUVAR&XNCPU,XMODEF2,XAPPLF2,XSPINIT,XENVNR,XSECP
// SETPFIX LIMIT=144K
// EXEC PROC=DTRCICST                  ASSGNS FOR CICS FILES
// EXEC PROC=DTRINFOA                  ASSGNS FOR INFO ANAL FILES
// EXEC PROC=DTRICCF                   ASSGN FOR DTSFILE
// ASSGN SYS005,UA
// ASSGN SYS006,UA
// ASSGN SYS007,UA
// ASSGN SYS008,UA
// ASSGN SYS009,SYSLOG
LOG
// ID USER=DBDCCICS
NOLOG
// EXEC DTSANALS                       RECOVER IF DTSFILE DESTROYED
RECOVER OPT
$$/*
```

Figure 16. Skeleton SKCICS, Part 1 of 2 (Starting Up the CICS Transaction Server and VSE/ICCF)

In a system with security (access control) active, the // ID statement for user DBDCCICS ensures that CICSICCF has the appropriate access rights to the control file. If you submit the job, your access rights are inherited by the CICSICCF startup job in the VSE/POWER reader queue. If your access rights as a user are adequate, it is recommended to delete the ID statement from the skeleton to avoid an exposure of the password. Chapter 7, "Protecting Resources," on page 129 describes the access control support in detail.

Instead of CICSICCF, you may use a name of your own for the startup job. To help you make a decision, read first "Considerations for Naming Conventions" on page 18. If you use your own name, you must also update procedures USERBG and COLDJOBS (skeletons SKUSERBG and SKCOLD) with the name you have chosen since they also call CICSICCF. With skeleton SKCOLD you can add your own jobs to the load list of procedure COLDJOBS. Note that these jobs must be cataloged in a VSE library.

For an explanation of the SETPFIX definition, refer to “Skeletons for Static Partition Allocations” on page 32.

For CLASS (in the $$$$ JOB statement) specify the identifier of the partition in which CICS is running. The skeleton assumes F2.

EOJMSG=YES is needed for an unattended node environment (it is included for compatibility reasons only since unattended nodes are no longer supported). At end-of-job (EOJ), VSE/POWER issues a message which initiates automated shutdown or the processing of other command lists.

Replace **F2** with the identifier of the partition if you run CICS with VSE/ICCF in another partition than F2.

```
*   WAITING FOR VTAM TO COME UP
// EXEC IESWAITT
$$/*
*   WAITING FOR TCP/IP TO COME UP
* // EXEC REXX=IESWAITR,PARM='TCPIP00'
$$/*
// OPTION SYSPARM='00'
// ASSGN SYS020,SYSLST
// SETPARM ELIM=14M
// IF XENVNR = B THEN
// SETPARM ELIM=25M
// IF XENVNR = C THEN
// SETPARM ELIM=200M
// IF XMODEF2 = COLD THEN
// GOTO COLDST
// SETPARM XMODEF2=AUTO
// GOTO STARTCIC
/. COLDST
// SETPARM XMODEF2=COLD
/. STARTCIC
// IF XSECP = RECOVERY THEN
// GOTO RECO
// EXEC DFHSIP,SIZE=DFHSIP,PARM='APPLID=&XAPPLF2.,START=&XMODEF2.,EDSAL*
               IM=&ELIM.,SI',DSPACE=2M,OS390
SIT=SP,STATRCD=OFF,MXT=20,NEWSIT=YES,
$$/*
// GOTO STAT
/. RECO
// EXEC DFHSIP,SIZE=DFHSIP,PARM='APPLID=&XAPPLF2.,START=&XMODEF2.,EDSAL*
               IM=&ELIM.,SI',DSPACE=2M,OS390
SEC=NO,SIT=SP,STATRCD=OFF,MXT=20,NEWSIT=YES,
$$$$ SLI MEM=DFH$SVEX.J,S=PRD1.BASE
$$$$ SLI MEM=IESVAEXC.Z,S=IJSYSRS.SYSLIB
$$/*
/. STAT
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF2=INACTIVE'
$$/*
$$/&
$$$$ EOJ
/+
CATALOG  LDCICS.PROC     REPLACE=YES DATA=YES
// EXEC DTRIINIT
   LOAD CICSICCF.Z
/*
/+
/*
// EXEC PROC=LDCICS        TO LOAD CICSICCF INTO RDR QUEUE
/&
$$ EOJ
```

Figure 17. Skeleton SKCICS, Part 2 of 2 (Starting Up the CICS Transaction Server and VSE/ICCF)

The application IESWAITT (called with // EXEC IESWAITT) must be defined to z/VSE as a VTAM application. This definition exists if you are using z/VSE as shipped by IBM. If this definition is removed because of modifications, you must use the *Maintain VTAM Application Names* dialog and redefine IESWAITT as application name (APPLID) to VTAM.

If there is a need to wait until TCP/IP is up, activate statement

```
* // EXEC REXX=IESWAITR,PARM='TCPIP00'
```

by removing the preceding asterisk and blank character provided that the TCP/IP startup job is TCPIP00.

Note that the default setting for SVA in the DFHSIT is YES. The exclude list IESVAEXC is used to allow a CICS coexistence environment.

After you have modified the skeleton, enter the following command from the editor's command line:

@DTRSEXIT

This command calls a macro that deletes specific comments from the skeleton. You should do this *before* filing the skeleton.

The skeleton includes additional $$ characters. They are needed to mask off VSE/POWER JECL statements. Program DTRIINIT, described in the manual *z/VSE System Utilities* under "DTRIINIT Utility", replaces the $$ characters with VSE/POWER JECL statements for cataloging.

Skeleton for Starting Up VTAM (SKVTAM)

The SKVTAM skeleton creates a startup job for VTAM running in default partition F3. The job stream created with SKVTAM catalogs the startup job (VTAMSTRT) into IJSYSRS.SYSLIB and loads it directly into the VSE/POWER reader queue.

If you have to perform a BASIC startup (because of VTAM startup problems, for example), perform the following steps:

- Perform BASIC startup.
- Use skeleton SKVTAM to create and catalog members VTAMSTRT.Z and LDVTAM.PROC into IJSYSRS.SYSLIB.
- Perform a normal startup.
- Run procedure LDVTAM.PROC to load the startup job VTAMSTRT into the VSE/POWER reader queue.
- Release startup job VTAMSTRT to continue normal startup.

```
* $$ JOB JNM=CATVTAM,DISP=D,CLASS=0
// JOB CATVTAM                 CATALOG VTAMSTRT AND LDVTAM, LOAD VTAMSTRT
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG VTAMSTRT.Z   REPLACE=YES
$$$$ JOB JNM=VTAMSTRT,DISP=L,CLASS=3
// JOB VTAMSTRT                            START VTAM
// OPTION DUMP,SADUMP=5
// SETPARM XNCPU=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF3=ACTIVE'
$$/*
// SETPFIX LIMIT=424K
* // SETPFIX LIMIT=(,300K)
// ASSGN SYS000,UA
// ASSGN SYS001,DISK,VOL=SYSWK1,SHR        TRACE FILE ASSIGNMENT
// ASSGN SYS004,DISK,VOL=SYSWK1,SHR        TRACE FILE ASSIGNMENT
// ASSGN SYS005,DISK,VOL=SYSWK1,SHR        NCP LOAD/DIAGNOSIS FILE ASSIGNMENT
```

Figure 18. Skeleton SKVTAM, Part 1 of 2 (Starting Up VTAM)

VTAMSTRT is the name of the job used by z/VSE for VTAM startup in default partition F3. You may use your own name instead. To help you make a decision, read first "Considerations for Naming Conventions" on page 18. If you use your own name, you must also update procedures USERBG and COLDJOBS (skeletons SKUSERBG and SKCOLD) with the name you have chosen since they also call

VTAMSTRT. With skeleton SKCOLD you can add your own jobs to the load list of procedure COLDJOBS. Note that these jobs must be cataloged in a VSE library.

For an explanation of the SETPFIX definition, refer to “Skeletons for Static Partition Allocations” on page 32.

For CLASS (in the $$$$ JOB statement), specify the identifier of the partition in which VTAM is running. The skeleton assumes F3.

For the // ASSGN statements, specify the VOLIDs of the disk devices where each file resides. SYS000 **must** be unassigned (UA) because VTAM uses it internally. The TRACE program addresses the TRACE file as SYS001. The TPRINT program addresses it as SYS004.

```
// LIBDEF  PHASE,SEARCH=(PRD2.COMM,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD2.COMM2,PRD2.CONFIG,PRD1.BASED,PRD1.BASE),PERM
// LIBDEF    OBJ,SEARCH=(PRD2.COMM,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD2.COMM2,PRD2.CONFIG,PRD1.BASED,PRD1.BASE),PERM
// LIBDEF SOURCE,SEARCH=(PRD2.COMM,
               YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,YYYYYYY.YYYYYYYY,
               PRD2.COMM2,PRD2.CONFIG,PRD1.BASED,PRD1.BASE),PERM
// LIBDEF   DUMP,CATALOG=SYSDUMP.F3,PERM
// EXEC ISTINCVT,SIZE=ISTINCVT,PARM='CUSTNO=Cxxx-xxx-xxxx,VTAMPW=yyyy-y*
               yyy-yyyy-yyyy-yyyy',DSPACE=2M
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF3=INACTIVE'
$$/*
$$/&
$$$$ EOJ
/+
CATALOG LDVTAM.PROC REPLACE=YES DATA=YES
// EXEC DTRIINIT
   LOAD VTAMSTRT.Z
/*
/+
/*
// EXEC PROC=LDVTAM              TO LOAD VTAM STARTUP INTO RDR QUEUE
/&
* $$ EOJ
```

Figure 19. Skeleton SKVTAM, Part 2 of 2 (Starting Up VTAM)

Replace the **Y** strings in each SEARCH chain. The LIBDEF statements define the permanent sublibrary search chains for phase, object, source, and dump library members.

Replace **F3** with the identifier of the partition if VTAM runs in another partition than F3.

After you make the changes, run the DTRSEXIT macro. This macro deletes specific comments from the file. You should do this before you file the skeleton. On the command line, enter:

@DTRSEXIT

The skeleton includes additional $$ characters. They are needed to mask off VSE/POWER JECL statements. Program DTRIINIT, described in the manual *z/VSE System Utilities* under “DTRIINIT Utility”, replaces $$ characters with VSE/POWER JECL statements for cataloging.

Skeleton for Starting Up TCP/IP (SKTCPSTR)

The SKTCPSTR skeleton creates a startup job for TCP/IP running in default partition F7. The job stream created with SKTCPSTR catalogs the procedure LDTCPIP into IJSYSRS.SYSLIB, which loads the startup job (TCPIP00) into VSE/POWER.

```
* $$ JOB JNM=CATTCPIP,DISP=D,CLASS=0
// JOB CATTCPIP CATALOG TCPIP00 AND LDTCPIP PROCEDURE LOAD TCPIP00
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG  TCPIP00.Z        REPLACE=YES
$$$$ JOB JNM=TCPIP00,DISP=L,CLASS=7,EOJMSG=YES
$$$$ LST CLASS=A,DISP=D,RBS=100
// JOB TCPIP00          TCP/IP  STARTUP
// ID USER=VCSRV
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// SETPFIX LIMIT=(400K)
// SETPFIX LIMIT=(,2100K)
// EXEC PROC=DTRICCF
// EXEC IPNET,SIZE=IPNET,PARM='ID=00,INIT=IPINIT00',DSPACE=4M
$$/*
$$/&
$$$$ EOJ
/+
CATALOG  LDTCPIP.PROC     REPLACE=YES DATA=YES
// EXEC DTRIINIT
   LOAD TCPIP00.Z
/*
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY TCPIP00.Z REP=YES
/*
// EXEC PROC=LDTCPIP      TO LOAD TCP/IP  INTO RDR QUEUE
/&
* $$ EOJ
```

Figure 20. Skeleton SKTCPSTR (Starting Up TCP/IP)

You should increase the SETPFIX value above 16 MB by 1 MB per OSAX link. Therefore, a default value of 2100 KB would allow for two OSAX links.

Replace **F7** with the identifier of the partition if TCP/IP runs in another partition than F7.

After you make the changes, run the DTRSEXIT macro. This macro deletes specific comments from the file. You should do this before you file the skeleton. On the command line, enter:

@DTRSEXIT

The skeleton includes additional $$ characters. They are needed to mask off VSE/POWER JECL statements. Program DTRIINIT, described in the manual *z/VSE System Utilities* under "DTRIINIT Utility", replaces $$ characters with VSE/POWER JECL statements for cataloging.

Skeleton for Loading User Jobs During a COLD Startup (SKCOLD)

During a COLD startup z/VSE processes procedure COLDJOBS. This procedure activates program DTRIINIT and provides a list of jobs to be loaded into the VSE/POWER reader queue. The loading of the jobs is done by program DTRIINIT. Refer to the manual *z/VSE System Utilities* under "DTRIINIT Utility" for further details about program DTRIINIT. This manual also points out what must be considered when loading jobs in a system with security (access control) active.

With skeleton SKCOLD you can add your own jobs to the load list of procedure COLDJOBS. Note that these jobs must be cataloged in a VSE library.

```
* $$ JOB JNM=CATALOG,CLASS=0,DISP=D
// JOB CATALOG
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG COLDJOBS.PROC   R=Y  DATA=YES
// EXEC DTRIINIT
  LOAD CICSICCF.Z
  LOAD STARTVCS.Z      LOAD CONNECTOR SERVER JOB
  LOAD TAPESRVR.Z      LOAD TAPE SERVER JOB
  LOAD CICS2.Z         LOAD CICS2; IF YOU DO NOT USE CICS2 DELETE IT
  LOAD CICSOLD.Z       LOAD CICSOLD; IF YOU DO NOT USE CICSOLD DELETE IT
/*
// EXEC DTRIINIT
 ACCESS S=PRD2.SCEEBASE    LOAD LE/VSE JOBS
 LOAD CEEWOPTJ.Z
 LOAD CEEWARC.Z
/*
// ID USER=DUMMY,PWD=DUMMY
// EXEC DTRIINIT
  LOAD VTAMSTRT.Z
  LOAD PAUSEBG.Z
  LOAD PAUSEFA.Z
  LOAD PAUSEFB.Z
  LOAD PAUSEF1.Z
  LOAD PAUSEF2.Z
  LOAD PAUSEF3.Z
  LOAD PAUSEF4.Z
  LOAD PAUSEF5.Z
  LOAD PAUSEF6.Z
  LOAD PAUSEF7.Z
  LOAD PAUSEF8.Z
  LOAD PAUSEF9.Z
  LOAD PRTDUMPA.Z
  LOAD PRTDUMPB.Z
  LOAD PRTDUC2A.Z    LOADED FOR CICS2; IF YOU DO NOT USE CICS2 DELETE IT
  LOAD PRTDUC2B.Z    LOADED FOR CICS2; IF YOU DO NOT USE CICS2 DELETE IT
  LOAD PRTDUCOA.Z    LOADED FOR CICSOLD; IF YOU DO NOT USE CICSOLD DELETE IT
  LOAD PRTDUCOB.Z    LOADED FOR CICSOLD; IF YOU DO NOT USE CICSOLD DELETE IT
/*
/+
/*
/&
* $$ EOJ
```

Figure 21. Skeleton SKCOLD (Loading Jobs into Reader Queue)

STARTVCS is the startup job for the Connector Server. TAPESRVR is the startup job for the Virtual Tape Data Handler.

In a system with security (access control) active, procedure COLDJOBS must be called with the appropriate access rights to load startup jobs CICSICCF and CICS2 into the VSE/POWER reader queue. The access rights are inherited by the jobs loaded. The // ID statement for user DUMMY turns off these access rights.

Refer to Chapter 7, “Protecting Resources,” on page 129 for a detailed description of the z/VSE access control support.

After you made the changes, run the DTRSEXIT macro. This macro deletes specific comments from the skeleton. You should do this before you file the skeleton. On the command line, enter:

```
@DTRSEXIT
```

Skeleton for Loading a Job (SKLOAD)

Skeleton SKLOAD catalogs a job into IJSYSRS.SYSLIB and loads the job (via procedure LDPAUSEC) into the VSE/POWER reader queue. Job PAUSEC is used as an example.

```
* $$ JOB JNM=CATPAUSE,DISP=D,CLASS=0
// JOB CATPAUSE           CATALOG PAUSEC.Z AND LDPAUSEC, LOAD PAUSEC
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG  PAUSEC.Z         REPLACE=YES
$$$$ JOB JNM=PAUSEC,DISP=L,CLASS=C,EOJMSG=YES
$$$$ LST CLASS=A,DISP=D
// JOB PAUSEC
// PAUSE
$$/&
$$$$ EOJ
/+
CATALOG LDPAUSEC.PROC     REPLACE=YES DATA=YES
// EXEC DTRIINIT
   ACCESS IJSYSRS.SYSLIB
   LOAD PAUSEC.Z
/*
/+
/*
// EXEC PROC=LDPAUSEC     TO LOAD PAUSEC INTO RDR QUEUE
/&
* $$ EOJ
```

Figure 22. Skeleton SKLOAD (Cataloging and Loading a Job)

Before you file the skeleton, run the DTRSEXIT macro. This macro deletes specific comments from the skeleton. On the command line, enter:

```
@DTRSEXIT
```

The skeleton includes additional $$ characters. They are needed to mask off VSE/POWER JECL statements. Program DTRIINIT, described in the manual *z/VSE System Utilities* under “DTRIINIT Utility”, replaces the $$ characters with VSE/POWER JECL statements for cataloging.

Skeleton for Tailoring $COMVAR Procedure (SKCOMVAR)

You must complete this skeleton if your environment includes at least two CPUs which share disk devices (DASD sharing). Procedure $COMVAR serves to identify the currently active CPU. The default $COMVAR is set to a single CPU environment (XNCPU=1).

Skeleton SKCOMVAR provides statements for three CPUs. Change, add, or delete statements as required. You can add statements for up to 31 CPUs. Replace the

- **X** string with the 12 character CPU ID of your second CPU, and the
- **Y** string with the 12 character CPU ID of your third CPU.

$COMVAR.PROC is cataloged into library IJSYSRS.SYSLIB and PRD2.SAVE.

The manual *z/VSE Guide to System Functions* provides further details about DASD sharing under “DASD Sharing with Multiple VSE Systems”.

```
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG $COMVAR.PROC DATA=YES R=Y
SETPARM XNCPU=1
// EXEC DTRISCPU,SIZE=AUTO,PARM='XXXXXXXXXXXX'    (12 CHARS CPUID2)
IF $RC=0 THEN
SETPARM XNCPU=2
// EXEC DTRISCPU,SIZE=AUTO,PARM='YYYYYYYYYYYY'    (12 CHARS CPUID3)
IF $RC=0 THEN
SETPARM XNCPU=3
/*
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY $COMVAR.PROC     REPL=YES
```

Figure 23. Skeleton SKCOMVAR (DASD Sharing Environment)

After you made the changes, run the DTRSEXIT macro. This macro deletes specific comments from the skeleton. You should do this before you file the skeleton. On the command line, enter:

```
@DTRSEXIT
```

Skeleton for Starting Up the Virtual Tape Server (SKVTASTJ)

The SKVTASTJ skeleton creates a startup job for the Virtual Tape Server partition and catalogs it into libraries IJSYSRS.SYSLIB and PRD2.SAVE.

The procedure LDVTA then loads the startup job (TAPESRVR) into the VSE/POWER reader queue.

```
* $$ JOB JNM=CATSTVTA,DISP=D,CLASS=0
// JOB CATSTVTA       CATALOG TAPESRVR AND LDVTA, LOAD TAPESRVR
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG  TAPESRVR.Z        REPLACE=YES
$$$$ JOB JNM=TAPESRVR,DISP=L,CLASS=R,LOG=NO
$$$$ LST CLASS=A,DISP=D,PURGE=0004
// JOB TAPESRVR START UP VSE TAPE SERVER
// ID USER=VCSRV
// OPTION SYSPARM='00'
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// EXEC $VTMAIN,SIZE=$VTMAIN
$$/*
$$/&
$$$$ EOJ
/+
CATALOG  LDVTA.PROC     REPLACE=YES DATA=YES
// EXEC DTRIINIT
   LOAD TAPESRVR.Z
/*
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY TAPESRVR.Z REP=YES
/*
// EXEC PROC=LDVTA        TO LOAD  TAPE SERVER  INTO RDR QUEUE
/&
* $$ EOJ
```

Figure 24. Skeleton SKVTASTJ (Starting Up Virtual Tape Server)

If you wish, you can replace **CLASS=R** with another class.

After you make the changes, run the DTRSEXIT macro. This macro deletes specific comments from the file. You should do this before you file the skeleton. On the command line, enter:

@DTRSEXIT

The skeleton includes additional $$ characters. They are needed to mask off VSE/POWER JECL statements. Program DTRIINIT, described in the manual *z/VSE System Utilities* under "DTRIINIT Utility", replaces $$ characters with VSE/POWER JECL statements for cataloging.

Skeleton for Starting Up VSE Connector Server (SKVCSSTJ)

The SKVCSSTJ skeleton creates a startup job for the VSE Connector Server partition and catalogs it into libraries IJSYSRS.SYSLIB and PRD2.SAVE.

The procedure LDVCS then loads the startup job (STARTVCS) into the VSE/POWER reader queue.

```
* $$ JOB JNM=CATSTVCS,DISP=D,CLASS=0
// JOB CATSTVCS       CATALOG STARTVCS AND LDVCS, LOAD STARTVCS
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG  STARTVCS.Z        REPLACE=YES
$$$$ JOB JNM=STARTVCS,DISP=L,CLASS=R
$$$$ LST CLASS=A,DISP=D
// JOB STARTVCS START UP VSE CONNECTOR SERVER
// ID USER=VCSRV
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// OPTION SYSPARM='00'
// EXEC IESVCSRV,PARM='DD:PRD2.CONFIG(IESVCSRV.Z)'
$$/*
$$/&
$$$$ EOJ
/+
CATALOG  LDVCS.PROC     REPLACE=YES DATA=YES
// EXEC DTRIINIT
   LOAD STARTVCS.Z
/*
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY STARTVCS.Z REP=YES
/*
// EXEC PROC=LDVCS        TO LOAD  VCS STARTUP  INTO RDR QUEUE
/&
* $$ EOJ
```

Figure 25. Skeleton SKVCSSTJ (Starting Up VSE Connector Server)

If you wish, you can replace **CLASS=R** with another class.

After you make the changes, run the DTRSEXIT macro. This macro deletes specific comments from the file. You should do this before you file the skeleton. On the command line, enter:

```
@DTRSEXIT
```

The skeleton includes additional $$ characters. They are needed to mask off VSE/POWER JECL statements. Program DTRIINIT, described in the manual *z/VSE System Utilities* under “DTRIINIT Utility”, replaces $$ characters with VSE/POWER JECL statements for cataloging.

Chapter 3. Modifying Predefined Environments

For a detailed description of the predefined environments provided by z/VSE, refer to the manual *z/VSE Planning* under “Predefined System Environments”.

This chapter discusses:

- Modifying library search chains
- Changing use of static partitions
- Modifying static partition allocations
- Moving to another environment
- Modifying dynamic partition support

For the task of “Moving the Label Area to a Virtual Disk”, refer to the manual *z/VSE Planning*.

Modifying Library Search Chains

You can modify the library search chains for partitions controlled by VSE/POWER:

1. Copy skeleton SKLIBCHN to your VSE/ICCF primary library. Refer to “Skeleton for Defining Library Search Chains (SKLIBCHN)” on page 52 for details about the skeleton.
2. In the skeleton, modify the LIBDEF statements by adding your sublibraries in the correct sequence.
3. Copy skeleton SKENVSEL to your VSE/ICCF primary library. Modify it to catalog your changes in skeleton SKLIBCHN. Refer to “Skeleton for Cataloging Startup Changes (SKENVSEL)” on page 31 for details about skeleton SKENVSEL.

Changing Use of Static Partitions

To have a program run in another static partition, proceed as follows:

1. In CPUVARn.PROC, set the system variables XUSEyy and XSTATyy as required for the partitions involved via program DTRSETP. The manual *z/VSE System Utilities* describes program DTRSETP under “DTRSETP Utility” in detail.

   If you use a different partition for a CICS system, you must also change the variable XAPPLyy in CPUVARn which defines the application name for a CICS system. yy defines the partition used.
2. If necessary, modify the JCL startup procedures and the startup job(s) involved. If possible, use the skeletons provided. Also, if necessary, delete obsolete startup jobs in the VSE/POWER reader queue and update procedure COLDJOBS via skeleton SKCOLD.
3. Modify the partition allocations as needed. If possible, use the skeletons provided. Refer to “Modifying Static Partition Allocations” on page 68 for details.
4. Modify the VSE/POWER autostart statements in the POWSTRTn procedure. Use skeleton SKPWSTRT.
5. If required, put PARSTD labels into the STDLABUS procedure. Use skeleton STDLABUS. “Creating Standard Labels for Non-VSE/VSAM User Files” on page 260 describes skeleton STDLABUS.

6. If required, modify or add library search chain. Use skeleton SKLIBCHN which is described under "Skeleton for Defining Library Search Chains (SKLIBCHN)" on page 52.
7. You may want to change VSE/POWER job classes. They are defined in the POWSTRTn procedure and in the startup jobs in the VSE/POWER reader queue. Use the appropriate skeletons (SKPWSTRT, SKCICS, and SKVTAM).
8. If required, ask the operator to enter a PRTY command or change the JCL procedures accordingly. This may be necessary since the priorities for the programs running in the partitions are now different from those specified in $0JCL or your own procedure.

Modifying Static Partition Allocations

Depending on your environment selected during initial installation (A, B, or C), you must use one of the following skeletons:

SKALLOCA = 12 static partitions (small system)
SKALLOCB = 12 static partitions (medium system)
SKALLOCC = 12 static partitions (large system)

You can vary the partition sizes within the boundaries of the VSIZE specified for the predefined environment selected. Refer to the manual *z/VSE Planning* under "Selecting a Predefined Environment" for the VSIZE values of the predefined environments. You can increase the VSIZE via the *Tailor IPL Procedure* dialog which may also mean that a change in the DPD definitions of the IPL procedure is required.

To change partition allocations, perform the following steps:

1. Copy the corresponding SKALLOCx skeleton to your primary VSE/ICCF library and modify the copied skeleton as needed. Refer to "Skeletons for Static Partition Allocations" on page 32 for details about these skeletons.

   If you change the procedure name ALLOC in the CATALOG statement, you must use the new name as input for skeleton SKJCL0. In the skeleton, change the EXEC PROC statement for the allocation procedure accordingly. Refer to "Skeletons for Starting Up BG Partition" on page 36 for details on skeleton SKJCL0.
2. If you change the default procedure name $0JCL in the CATALOG statement of SKJCL0, you must specify the new name also in the $ASIPROC master procedure (if you created one) or the operator must enter it during IPL.
3. Copy skeleton SKENVSEL to your primary VSE/ICCF library and modify it to catalog your changes.

   Refer to "Skeleton for Cataloging Startup Changes (SKENVSEL)" on page 31 for details on skeleton SKENVSEL.
4. Submit the job stream created for processing.

Notes:

1. You can change the partition size temporarily with the ALLOC command. Refer to the manual *z/VSE System Control Statements* under "ALLOC" for details about this command.
2. For the changes introduced with VSE/ESA 1.3 concerning ALLOC R, refer to "Skeletons for Static Partition Allocations" on page 32.

Moving to Another Environment

Moving to another environment involves a number of tailoring tasks. These vary according to your special system requirements. Consider the following as a guideline that helps you define the tailoring tasks to be performed for creating a specific environment.

Moving from Predefined Environment A to B/C, or from B to C

The predefined environments provided are described in detail in the manual *z/VSE Planning* under "Predefined System Environments".

To move from predefined environment A to predefined environment B or C, proceed as follows:

- For an environment with a larger VSIZE, use the *Tailor IPL Procedure* dialog to also define new DPD extents. The dialog gives you hints for how much space you need. Use the *Tailor IPL Procedure* dialog (which is described under "Tailoring the IPL Procedure" on page 8) also if you want to change other IPL parameters.
- To change existing startup procedures, use the skeletons provided. Copy the skeletons first to your VSE/ICCF primary library.
- Set environment number XENVNR in CPUVAR1.PROC by running program DTRSETP in the BG partition:

  ```
  // EXEC DTRSETP,SIZE=AUTO,PARM='CPUVAR1;;SET XENVNR=n'
  ```

  where n is the environment number: A, B, or C.

  Refer to *z/VSE System Utilities* under "DTRSETP Utility" for details about how to use program DTRSETP.
- Change partition allocations as required by using skeleton SKALLOCn.

Moving to an Environment of Your Own Design

If you want to define an environment of your own design, proceed as follows:

1. Use the *Tailor IPL Procedure* dialog to modify IPL parameters such as virtual storage (VSIZE) and page data set extents (DPD). For details about the dialog, refer to "Tailoring the IPL Procedure" on page 8.
2. Depending on your requirements you may use one or more of the skeletons provided by z/VSE to tailor your startup. For partition allocations, consider using skeleton SKALLOCA as a sample. Refer to "Using Skeletons for Tailoring System Startup" on page 30 for details about the skeletons provided.
3. If needed, create or change startup jobs for the partitions. A startup job must be available as cataloged procedure in a VSE library. Load the startup jobs into the VSE/POWER reader queue. Use program DTRIINIT which is described in detail in the manual *z/VSE System Utilities* under "DTRIINIT Utility".
4. Delete obsolete startup jobs in the VSE/POWER reader queue and update procedure COLDJOBS (skeleton SKCOLD) if necessary.

Modifying the Dynamic Partition Support

If you plan to use dynamic partitions, you should first consult the manual *z/VSE Planning* under “Planning for Dynamic Partition Support”. It discusses planning aspects to be considered before using this support.

Notes:

1. The maximum number of dynamic classes per table is **23**.
2. The maximum number of dynamic class tables is **36**.

Predefined environments A, B, and C provide dynamic partition support. If you want to modify this support or create a dynamic partition environment of your own, follow the steps outlined below. All steps can be performed while your system is running and are described in detail on the following pages.

1. Tailor the IPL Procedure via the *Tailor IPL Procedure* dialog.
2. Catalog the JCL startup procedure for a dynamic partition class via skeleton SKJCLDYN.
3. Tailor the VSE/POWER startup procedure via skeleton SKPWSTRT to have a specific dynamic class table be loaded automatically during startup.
4. Define one or more dynamic class tables via the *Maintain Dynamic Partitions* dialog according to your requirements.

You can activate a dynamic class table with the PLOAD command of VSE/POWER or have it activated automatically during startup (step 3).

Tailoring the IPL Procedure

Use the *Tailor IPL Procedure* dialog (as described under “Tailoring the IPL Procedure” on page 8) if you want to change IPL parameters such as:

- NPARTS (IPL SYS command)

  Defines the total number of partitions (static and dynamic) you want to use. You can increase the number of dynamic partitions but not of static partitions (where the maximum is 12). The maximum number of dynamic partitions possible depends on the user environment. A theoretical value is 150 - 200.
- VSIZE (IPL supervisor parameters command)

  If you want to increase this value, you must also increase the values in the IPL DPD commands for the page data set extents. Consult the manual *z/VSE Planning* under “z/VSE Disk Layouts” for the layout of DOSRES and SYSWK1 on the various disk device types.

  Note: You must ensure that no overlap occurs with other files when enlarging or relocating the page data set extents.

Depending on the number of the dynamic partitions you are going to use, there may also be a need to increase the GETVIS area. GETVIS is a parameter of the IPL SVA command. To activate your IPL changes, you must re-IPL the system.

Cataloging JCL Startup Procedures

For each dynamic class defined in the dynamic class table you must also define a startup procedure. This procedure is processed each time a dynamic partition is created. You may use the same procedure for more than one dynamic class. To catalog such a procedure, use skeleton **SKJCLDYN** which includes a sample profile. The layout of the skeleton SKJCLDYN is shown below.

```
CATALOG STDPROF.PROC  DATA=YES REPLACE=YES
// LIBDEF DUMP,CATALOG=SYSDUMP.DYN,PERM   ASSIGN SYSDUMP FOR DYN.PART.
// OPTION NODUMP
// EXEC PROC=LIBDEF      DEFINE THE PERMANENT LIBRARY SEARCH CHAIN
// SETPFIX LIMIT=48K
ASSGN SYSIN,FEC
ASSGN SYSPCH,FED
ASSGN SYSLST,FEE
ASSGN SYSLNK,DISK,VOL=DOSRES,SHR                SYSTEM LINK FILE
ASSGN SYS001,DISK,VOL=SYSWK1,SHR                SYSTEM WORK FILE 1
ASSGN SYS002,DISK,VOL=SYSWK1,SHR                SYSTEM WORK FILE 2
ASSGN SYS003,DISK,VOL=SYSWK1,SHR                SYSTEM WORK FILE 3
ASSGN SYS004,DISK,VOL=SYSWK1,SHR                SYSTEM WORK FILE 4
   .                     DEFINE YOUR OWN ASSIGNS IF NEEDED
   .                     ADD DLBL AND EXTENT STATEMENTS HERE IF NEEDED
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY STDPROF.PROC  REPLACE=YES
```

Figure 26. Skeleton SKJCLDYN

You find the name of the procedure (STDPROF) under PROFILE in Figure 27 on page 72. For details about procedure LIBDEF refer to "Skeleton for Defining Library Search Chains (SKLIBCHN)" on page 52.

After you have modified skeleton SKJCLDYN, enter the following command from the editor's command line:

@DTRSEXIT

This command calls a macro which deletes specific comments from the file. You should do this before filing the skeleton. Use skeleton SKENVSEL (shown under "Skeleton for Cataloging Startup Changes (SKENVSEL)" on page 31) for cataloging your changes.

Tailoring VSE/POWER Startup Procedure

Activate or modify the PLOAD command in skeleton SKPWSTRT to have the required dynamic class table automatically loaded during startup. For details about skeleton SKPWSTRT refer to "Skeleton SKPWSTRT" on page 45.

Activating a Dynamic Class Table

This step is **not** required if the operator will activate the dynamic class table at the console via the VSE/POWER command:

```
PLOAD DYNC,ID=n,FORCE
```

n identifies the table and FORCE ensures that even in case of wrong class definitions those classes which are correctly specified are activated.

Defining Dynamic Class Tables

z/VSE allows you to define up to 36 dynamic class tables. The name of the predefined table shipped is DTR$DYNC.

To define dynamic class tables, you use the *Maintain Dynamic Partitions* dialog. The dialog updates members DTR$DYNn.Z in system library IJSYSRS.SYSLIB when pressing PF5 on the panels shown later. For a detailed description of member DTR$DYNn.Z, refer to the manual *z/VSE Planning* .

To access the dialog, start with the *z/VSE Function Selection* panel and select:
2 (Resource Definition)
7 (Maintain Dynamic Partitions)

The dialog identifies invalid entries by displaying 2 or 5 in the OPT column of the panel shown in Figure 27. This may also happen when the dialog is reactivated and validity checking on a higher level is possible.

```
Administrator
Fast Path:   27
```

```
Synonym
Default:    ____   Yours:
```

You get first panel TAS$DYNA listing the dynamic class tables defined for the system. With this panel you can add, alter or delete dynamic class tables. You can define up to **36** dynamic class tables with up to **23** dynamic classes per table. If you enter option 1 (to add a new table) next to an existing table, the specifications of the dynamic classes are used as default for the new table. First, you must define a name (DTR$DYNX in the example) for the new table in panel TAS$DYN3 before panel TAS$DYN1 is displayed.

```
TAS$DYN1                 MAINTAIN DYNAMIC PARTITIONS

Enter option(s) and press enter.

                                          Dynamic class table : DTR$DYNX
 OPTIONS:   1 = ADD            2 = ALTER             5 = DELETE
 OPT   DYNAMIC  ENABLED   MAX NO. OF    STORAGE    MAXIMUM   DYNAMIC   PROFILE
        CLASS   1=YES     PARTITIONS  ALLOCATION   PROGRAM   SPACE
                2=NO                               SIZE      GETVIS
  _        C       1          9             1M       500K    128K     STDPROF
  _        P       1         32             1M       512K    128K     PWSPROF
  _        R       1          3             8M      1024K    128K     STDPROF
  _        S       1          2            15M      1024K    128K     STDPROF
  _        Y       1          8             3M      1024K    128K     STDPROF
  _        Z       1          3             5M      1024K    128K     STDPROF




PF1=HELP       2=REDISPLAY  3=END                     5=PROCESS
```

Figure 27. Maintain Dynamic Partitions (TAS$DYN1)

If you enter option 1 (in panel TAS$DYN1) to add a new class definition you get the panel shown in Figure 28 on page 73. The system displays default values if you selected an empty line. If you selected an already defined class, as in the example, the dialog uses this class and its parameters as a model for the new class.

```
TAS$DYN2              MAINTAIN DYNAMIC PARTITIONS

Enter the required data and press ENTER.

DYNAMIC CLASS..........               Enter one of the classes C - E, G - Z
NUMBER OF PARTITIONS...  8            Enter a number between 1 and 32
STORAGE ALLOCATION.....  2M           Specify in M bytes
MAXIMUM PROGRAM SIZE...  1024K        Specify in K or M bytes
DYNAMIC SPACE GETVIS...  128K         Specify in K bytes
ENABLED................  1            1=YES, 2=NO
PROFILE................  STDPROF      Name of the JCL procedure
NUMBER OF LOGICAL UNITS  50           Enter a number between 10 - 255

READER DEVICE..........  FEC
LIST OF PRINTER DEVICES  FEE   ___   ___   ___   ___   ___   ___
                         ___   ___   ___   ___   ___   ___   ___

LIST OF PUNCH DEVICES..  FED   ___   ___   ___   ___   ___   ___
                         ___   ___   ___   ___   ___   ___   ___

PF1=HELP      2=REDISPLAY 3=END
```

Figure 28. Maintain Dynamic Partitions (TAS$DYN2)

Description of parameters:

DYNAMIC CLASS
Specifies the class to which a dynamic partition belongs. Except for the letters A, B, and F you can use all letters of the alphabet.

NUMBER OF PARTITIONS
Specifies the maximum number of partitions belonging to the same class.

STORAGE ALLOCATION
Specifies the virtual storage available for a dynamic partition. The allocation includes the dynamic space GETVIS area:

```
storage allocation = partition allocation + dynamic space GETVIS area
```

The maximum partition allocation calculates as follows:

```
2GB - size of shared areas - size of dynamic space GETVIS area
```

2GB is the maximum address space size possible. The shared areas include shared partitions, system GETVIS area, and the area in which the supervisor resides.

MAXIMUM PROGRAM SIZE
Specifies the amount of contiguous virtual storage in a dynamic partition reserved for program execution.

DYNAMIC SPACE GETVIS
Specifies the size of the dynamic space GETVIS area of a partition. This area can be considered as an extension of the system GETVIS area. In the chapter "Storage Management" of the manual *z/VSE Guide to System Functions* you find details about the layout of dynamic partitions including GETVIS areas.

ENABLED
Specifies whether the dynamic class is to be enabled when a PLOAD command for activating the dynamic class table is processed by VSE/POWER.

PROFILE
Specifies the name of the JCL startup procedure which is to be processed each time VSE/POWER activates (creates) a dynamic partition of this class.

NUMBER OF LOGICAL UNITS
Specifies the maximum number of programmer logical units which can be allocated to each dynamic partition of this class.

READER DEVICE
Specifies the address of the spooled reader device for the dynamic partitions of this class.

LIST OF PRINTER DEVICES
Specifies the address(es) of the spooled printer device(s) for the dynamic partitions of this class.

LIST OF PUNCH DEVICES
Specifies the address(es) of the spooled punch device(s) for the dynamic partitions of this class.

Chapter 4. Configuring Non-Communication Devices

Introduction

During initial installation, you completed hardware configuration after signing on with the POST user ID. This is described in detail in the manual *z/VSE Installation* under "Installation Part 3 - Native and VM".

However, you can add, change, or delete devices on your system at any time. Use the *Configure Hardware* dialog to add or delete hardware addresses and to specify device characteristics.

By using the dialog as described below, you can create a list that shows all the devices (and addresses) which are part of your system. Such a **configuration list** helps you control the hardware attached to your system. You should create a new list whenever a hardware change is implemented. To create a configuration list for your installation, perform the following steps:

1. Start with the *z/VSE Function Selection* panel and select fast path 241. You get the panel *Hardware Configuration: Unit Address List*.
2. Press PF9 (PRINT). On the panel displayed you can select the type of configuration list you want; SNA or non-SNA terminal list, for example. After selecting one or more lists, press ENTER.
3. The configuration list(s) created are stored as library member **CONFLIST** in your VSE/ICCF primary library.
4. You can print the library member by selecting option 3 (PRINT) in the FULIST display of your primary library. The output is placed in the VSE/POWER list queue for printing.

This chapter describes how to use the *Configure Hardware* dialog for configuring **non-communication devices** such as disk devices, tapes, or printers.

For configuring **communication devices** like Open Systems Adapters devices (OSA Express, OSA-2), communication controllers, display stations, and personal computers, refer to the manual *z/VSE Networking Support*.

Selecting the Configure Hardware Dialog

Start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
4 (Hardware Configuration and IPL)
1 (Configure Hardware)

Administrator	Synonym	
Fast Path: 241	Default: ____	Yours:

You get the panel *Hardware Configuration: Unit Address List*. It is shown in Figure 29.

```
ADM$HDWB         HARDWARE CONFIGURATION: UNIT ADDRESS LIST



OPTIONS: 2 = ALTER DEVICE TYPE CODE/MODE      3 = SELECT FOR FURTHER PROCESSING
         4 = LIST SIMILAR DEVICES             5 = DELETE A DEVICE

    OPT     ADDR  DEVICE   DEVICE-TYPE   DEVICE SPECIFICATION  DEVICE  DEF
                                CODE           MODE             DOWN  INCOMPL
     _      009   CONSOLE    3277
     _      00E   PRT1
     _      102   3390-2     ECKD
     _      103   3390-2     ECKD
     _      104   3390-2     ECKD                                X
     _      105   3390-2     ECKD                                X
     _      180   3480       3480            00
     _      181   3480       3480            00
     _      2D0   FBA-SCSI   FBA

 POSITION NEAR ADDR == > ___
PF1=HELP       2=REDISPLAY  3=END                    5=PROCESS    6=ADD ADDR
PF7=BACKWARD   8=FORWARD    9=PRINT
```

Figure 29. Unit Address List of Hardware Configuration Dialog

The unit address list consists of one or more panels. It shows all I/O addresses (cuu) and the related devices as defined for your z/VSE system.

- An ’X’ in the column DEVICE DOWN indicates for a tape or disk device that this device is not available. By selecting 3 (SELECT FOR FURTHER PROCESSING), this status can be changed.
- An ’X’ in the column DEF INCOMPL (definition incomplete) indicates that you should specify additional details for that particular I/O address.

Various options and PF-key functions allow you to maintain your hardware configuration. They are listed below.

OPTIONS:

2 = ALTER DEVICE TYPE CODE/MODE

Select option 2 if you want to change the Device Type Code, or the Device Specification Mode.

3 = SELECT FOR FURTHER PROCESSING

Select option 3 if you want to change or add device characteristics other than Device Type Code or Device Specification Mode.

4 = LIST SIMILAR DEVICES

Select option 4 if you want only devices displayed that belong to a particular group. All disk devices or tapes for example.
5 = DELETE A DEVICE

Select option 5 if you want to delete an address (device) from the I/O address list.

POSITION NEAR ADDR:

This selection allows you to enter an address to position the FULIST close to this address. You can use 0 or FFF to skip to the top or to the bottom of the fulist. The input in this field is ignored when you press a PF key.

PF Keys:
5=PROCESS

Press PF5 if you have done all your changes using the options 2, 3, or 5, or PF6.
6=ADD ADDR

With PF6 you add a new address (device) to your hardware configuration. Depending on the type of device, several panels are displayed. You have to select the device you want to add to your installation and enter all device specific information required.
9=PRINT

Use PF9 to get a printout of the I/O address list. The output is stored as library member **CONFLIST** in your VSE/ICCF primary library.

Adding a Non-Communication Device

For most device parameters the system creates and uses defaults. But you must at least know the **device type** and **I/O address(es)** before you can add a new device.

Let us assume that an IBM tape device of type 3490 is to be added at address 182. The following steps are required:

1. Use fast path **241** from the *z/VSE Function Selection* panel. This gives you the first page of *Hardware Configuration: Unit Address List*. Figure 29 on page 76 shows this panel.
2. Press **PF6**. You get the panel *Hardware Configuration: Add a Device*. Enter the address (range) of the device and the device name (3490). Instead of entering the device name, you can enter a ’?’ to display the selection panel for the *Device Groups*. In this panel, enter:

   9 (Tape Units)

   Press **ENTER**.
3. You get the *Selection List: Devices* panel showing all tape devices supported by z/VSE. Select the correct device type:

   1 (3490)

   Press **ENTER**.

   The system redisplays panel *Hardware Configuration: Unit Address List* with the newly added device.
4. Press **PF5** to process (catalog) the updated hardware configuration. You get the panel *Hardware Configuration: Catalog Startup Members* shown in Figure 30. In this panel, those startup members are marked by an ’X’ which are affected by the change. In our example: IPL Procedures.

   Press **ENTER**.

5. You get the *Job Disposition* panel to submit the job to batch, file it in your VSE/ICCF primary library, or both.

```
 ADM$CRE1        HARDWARE CONFIGURATION: CATALOG STARTUP MEMBERS



Press ENTER to catalog the objects marked by an X. You may add or delete
an X as needed.


              X      IPL Procedures
              _      VTAM Book with Startup Options
              _      VTAM Books for Model Terminal Support
              _      VTAM Book for Local Non-SNA Terminals
              _      VTAM Book Local SNA Terminals
              _      VTAM Books for ICA attached Terminals
              _      VTAM Books for OSA or 3172 attached Terminals
              _      CICS CSD Group for terminals - VSETERM1
              _      CICS CSD Group for terminals - VSETERM2
              _      CICS CSD Group for terminals - VSETERM3



 PF1=HELP       2=REDISPLAY  3=END

 IPLPROC          SOURCE CREATED.
```

Figure 30. Panel for Cataloging Startup Members (Hardware Configuration)

Device Considerations

For most non-communication devices you only have to select the device type and enter the *cuu* address. The other device characteristics are known to the system. For disk devices and the IBM 3820 printer further panels prompt you for additional device characteristics. For tape devices, you can change and define additional characteristics with the SETMOD attention command.

You can also change the mode using option 2 (ALTER DEVICE TYPE CODE/MODE) on the *Hardware Configuration: Unit Address List* panel.

Disk Devices (Including FBA-SCSI Disks)

It is recommended that you initialize a disk device before you add it to your system. Under “Initializing Disks and Placing the VTOC”, the manual *z/VSE Installation* has details on how to initialize disk devices.

When you enter a disk device type, you get the panel *Hardware Configuration: Disk List* showing the disk devices specified and two columns with the options SHARED and DEVICE DOWN.

- SHARED

  Enter an **X** for those disk devices you want to share across systems. The following types of IBM disk devices can be used for (DASD) sharing:

  3380
  3390
  FBA
  FBA-SCSI

- DEVICE DOWN

  Enter an **X** for those disk devices that are not available (or should not be available) for operation.

  In a shared environment in which the volume labels are not unique, DEVICE DOWN may be required to prevent the system from accessing the wrong device when being addressed by volume label (VOLID).

> **Configuring FBA-SCSI Disks**
>
> The procedure for configuring FCP-attached SCSI disks (FBA-SCSI disks) is described in the chapter "Using SCSI Disks With Your z/VSE System" in the manual *z/VSE Planning*, SC33-8221. This procedure is based upon a practical example that covers all aspects of configuring for SCSI use.

Tape Devices

For the IBM 3480 and IBM 3490 the *Configure Hardware* dialog offers two choices for device definition: one with data compaction, one without. For details about data compaction, refer to the ADD command in the manual *z/VSE System Control Statements* under "ADD".

For the IBM 3590 the *Configure Hardware* dialog offers three choices for device definition: with 128, 256, or 384 track capacity. For the IBM 3592 the *Configure Hardware* dialog offers the 512 track capacity.

With option 2 (ALTER DEVICE TYPE CODE/MODE) the default mode setting can be changed. With option 3 (SELECT FOR FURTHER PROCESSING) the status of a tape device can be set to DEVICE DOWN (and reset).

Automated Tape Library Support

When using the IBM 3494 tape library, it is necessary to request the automated tape library support in the IPL SYS command with the parameter ATL.

You can use the *Tailor IPL Procedure* dialog for setting ATL.

IBM 3820 Printer

When you specify an IBM 3820 printer you have to define VTAM parameters associated with the printer. You define such parameters via the *Hardware Configuration: SNA Logical Unit List* panel.

You must provide the VTAM parameter **LOGAPPL**; for the parameters LOCAL ADDRESS, VTAM PARM TABLE, and LUNAME the system provides defaults.

Support of AFP Printers

You define printers for Advanced Function Printing (AFP) as any other printer via the *Configure Hardware* dialog. The IBM 3800-3, IBM 3825, and IBM 3827 are examples of such printers. To make use of their advanced functions, these printers require in addition the optional program PSF/VSE (Print Services Facility/VSE). z/VSE provides procedures and skeletons which support the installation and use of PSF/VSE. An example is the VSE/POWER startup procedure POWSTRTn which you can modify via skeleton SKPWSTRT. For details refer to "Skeleton SKPWSTRT" on page 45.

Virtual Disk for Label Area

As shipped, z/VSE provides a virtual disk with address FDF for holding the label area.

Considerations for Dummy Devices

You can change but not delete the following VSE/POWER and VSE/ICCF dummy devices: FED, FEE, FEF, FFD, FFE.

You cannot change or delete the following VSE/POWER and VSE/ICCF dummy devices: FEC, FFA, FFC.

You cannot change or delete the dummy device FFF which is a place holder for a dedicated **system console**.

Changing or Deleting a Non-Communication Device Specification

Use panel *Hardware Configuration: Unit Address List*. You get to it by selecting fast path **241** from the *z/VSE Function Selection* panel.

Enter **2** (Alter Device Type Code/Mode) in the option column if you want to change the device type code or mode of a device. Enter **5** (Delete a Device) if you want to delete a device.

Enter **3** (Select for Further Processing) in the option column if you want to change other device characteristics. Note that the following non-communication devices have characteristics that can be changed with option **3**:

- Disk devices.
- Tape devices
- IBM 3820 printer.

After you have changed device characteristics or selected deletion, proceed as shown in steps 4 and 5 for “Adding a Non-Communication Device” on page 77.

Chapter 5. Configuring Your System to Use SCSI Disks

This chapter describes how you configure your z/VSE system to use Fibre-Channel-attached SCSI (Small Computer System Interface) disks.

It contains these main sections:

- "Configuring FCP Adapters, SCSI Disks, and Connection Paths" on page 82
- "Using Multipathing to Access SCSI Disks" on page 90
- "Using Shared SCSI Disks" on page 91
- "Using the Attention Routine OFFLINE / ONLINE Commands" on page 92
- "Performing an IPL of z/VSE From a SCSI Disk" on page 92
- "Errors That Might Occur During Configuration" on page 95

Related Sections:

- "Changing Startup When Lock File Is Stored On SCSI DASD" on page 28
- The *z/VSE Planning*, SC33-8221 contains details of:
  - What the SCSI disk support provides you with (including a discussion of multipathing and DASD sharing).
  - The hardware and software pre-requisites for using SCSI disks.
  - The characteristics of a SCSI disk (including allowed disk sizes, and FBA CCW commands not supported).
  - How to use SCSI disks under z/VM.
  - What you must consider if you wish to use VSAM files on SCSI disks.
  - The space requirements for using SCSI as a system disk, including details of the FBA disk layouts of DOSRES und SYSWK1.
  - The GETVIS storage requirements when using SCSI disks.

Please Note That!

- The description in this section is based upon the use of the IBM ESS disk controller.
- Devices that have an FCP interface require an FCP switch.
- The term *FBA-SCSI disk* is also used to refer to a SCSI disk.

Configuring FCP Adapters, SCSI Disks, and Connection Paths

This section is based upon the example configuration of Figure 31.

Example of a SCSI Environment

The example shown in Figure 31 and described in "Configuring FCP Adapters, SCSI Disks, and Connection Paths," provide a practical example of how SCSI disks might be attached to a z/VSE host.

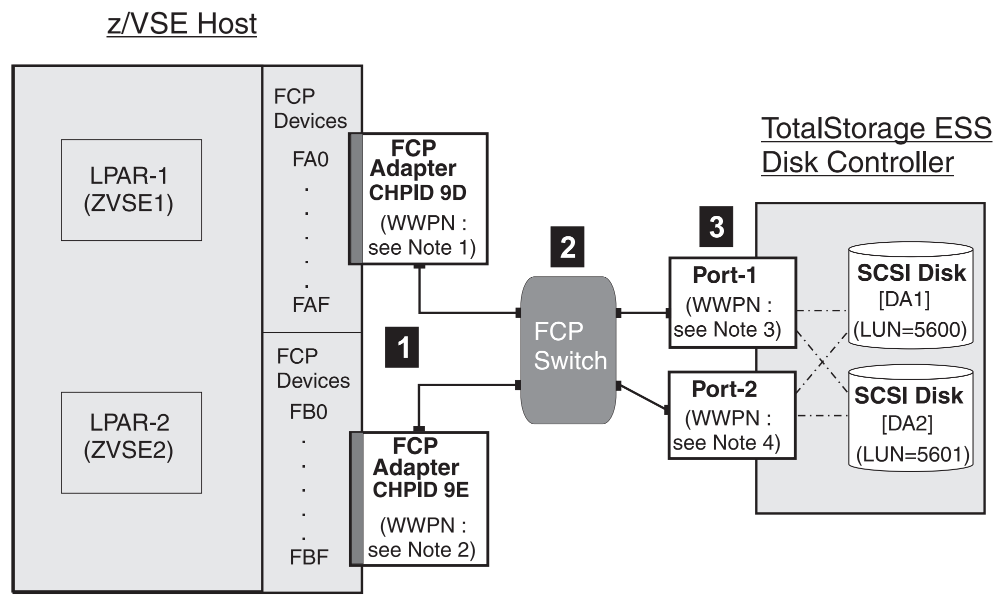


Figure 31. Example of a SCSI Environment

The example configuration shown in Figure 31 consists of:

1 Two physical zSeries FCP adapters with CHPIDs (channel path IDs) `9D` and `9E`. These physical FCP adapters (CHPIDs) are accessed as devices of type FCP that have been configured using the IOCP (Input/Output Configuration Program). Both physical FCP adapters (CHPIDs) are shared by the z/VSE systems running in LPAR-1 and LPAR-2. This is shown in the IOCP statements of Figure 32 on page 83. The physical FCP adapters can reside on the same FCP card or on different FCP cards (this is discussed in "Using Multipathing to Access SCSI Disks" on page 90).

The number of FCP devices that can be defined for each physical FCP adapter depends upon the hardware you are using. In Figure 31:

- Sixteen FCP devices have been defined for physical FCP adapter (CHPID 9D) in the range `FA0` to `FAF`.
- Sixteen FCP devices have been defined for physical FCP adapter (CHPID 9E) in the range `FB0` to `FBF`.

Note: Within one z/VSE system, one FCP device per physical FCP adapter is sufficient for accessing *both* the FBA-SCSIs `DA1` and `DA2`.

2 Each physical zSeries FCP adapter is connected to the FCP switch (which might be an IBM 2109 switch) via a physical cable.

3 The first port (Port-1) on the ESS controller has the worldwide port number (WWPN) 5005076300CA9A76. The second port (Port-2) on the ESS controller has the WWPN 5005076300C29A76. Both ports are connected to the FCP switch via physical cables. Ports are configured using the configuration software provided with the ESS disk controller, as described in "Configure the SCSI Disks in the ESS Controller."

A WWPN is a unique 64-bit string (16 hexadecimal numbers) that represents a port.

The ESS controller contains two SCSI disks with the logical unit numbers (LUN) 5600 and 5601. A LUN represents a SCSI device. The LUNs have been configured so they can be accessed via both FCP adapters. LUNs are configured using the configuration software provided with the ESS disk controller, as described in "Configure the SCSI Disks in the ESS Controller." The cuu addresses DA1 and DA2 are only used by z/VSE and are not part of the configuration process for the ESS controller.

Configure the FCP Adapters Using IOCP

Each FCP adapter that you plan to use with a zSeries processor must be configured in the IOCP (Input/Output Configuration Program).

An FCP adapter is identified in the zSeries I/O configuration by its channel path identifier (CHPID). The channel type for an FCP adapter is FCP. For the example shown in Figure 31 on page 82, you would use these type of statements:

```
   ⋮
CHPID PATH=(9D),SHARED,                                         *
      PARTITION=((ZVSE1,ZVSE2),(=)),TYPE=FCP
CHPID PATH=(9E),SHARED,                                         *
      PARTITION=((ZVSE1,ZVSE2),(=)),TYPE=FCP
   ⋮
CNTLUNIT CUNUMBR=0FA0,PATH=(9D),UNIT=FCP
CNTLUNIT CUNUMBR=0FB0,PATH=(9E),UNIT=FCP
   ⋮
IODEVICE ADDRESS=(FA0,016),UNITADD=00,CUNUMBR=(0FA0),UNIT=FCP
IODEVICE ADDRESS=(FB0,016),UNITADD=00,CUNUMBR=(0FB0),UNIT=FCP
```

Figure 32. IOCP Statements Used For Configuring FCP Adapters

Note: SCSI disks are *not* configured using the IOCP. Instead, the configuration programs supplied with the IBM TotalStorage ESS controller are used.

Configure the SCSI Disks in the ESS Controller

Notes:

1. This section provides you with an overview of the actions you must perform using the *ESS Specialist* (the configuration program of your ESS controller), based upon the example provided. For full details, refer to the documentation provided with your ESS Controller.
2. Please be aware that the information in this section may not exactly match the version of the ESS Specialist you are currently using.

Step 1. **Define an FCP adapter**.
For each of the two FCP adapters shown in Figure 31 on page 82, you must:

a. From the main menu, select **Storage Location**, then **Open System Storage**, and then **Modify Host Systems**. Now you can enter a **Nickname**. For **Host-Type**, you must select RS/6000 or Linux. The Host-Type will determine the format of the LUN (logical unit number) to be generated.

b. In the *Host Attachment* selection, select **Fibre Channel Attached**.

c. In the field *Worldwide Port Name*, you must specify the WWPN of your FCP adapter (CHPID) (for the example, either `5005076300C295A5` or `5005076300C695A5`). A WWPN is automatically assigned to each physical FCP adapter (CHPID) when it is shipped from the factory. You can obtain the WWPN by displaying the CHPID information using the Service Element.

d. In the *Fibre Channel Ports* selection, select the ESS controller's ports that can be accessed by the FCP adapter. You can either select all installed ports, or you can select specific ports. In the example provided, you would select Port-1 and Port-2 of Figure 31 on page 82.

e. Perform a configuration update.

Step 2. **Configure the ESS Ports.**
For each of the two ports (Port-1 and Port-2) shown in Figure 31 on page 82, you must:

a. From the main menu, select **Storage Location**, **Open System Storage**, and then **Configure Host Adapter Ports**.

b. Select the FCP Host Adapter port that you wish to configure.

c. From the *FC Ports Attributes* selection, you select:
   1) **Fibre Channel Topology**, and then **Point to Point (Switched Fabric)**.
   2) **Fibre Channel Protocol**, and then **FCP (Open Systems)**.

d. Perform a configuration update.

Step 3. **Define the LUNs**.

a. From the main menu, select **Storage Location**, **Open System Storage**, and then **Add Volumes**.

b. Select **Host** and then the **Nickname** you specified for your physical FCP adapter (entered in Step 1 above).

c. Select **Adapter**.

d. Select a Disk Group that contains free space.

e. Select a volume size, and then the number of volumes (SCSI disks) you wish to define for this size.

f. Perform a configuration update.

Step 4. **Specify Access to the LUNs**.

a. From the main menu, select **Storage Location**, **Open System Storage**, and then **Modify Volume Assignments**.

b. Select the volumes you wish to modify. In the example provided, this would be `5600` and `5601`.

c. Click **Assign selected volumes to target hosts**, **Use same ID/LUN in source and target**, and then **Select Target Hosts**. Select the Nicknames of your physical FCP adapters (entered in Step 1 above). This will allow the FCP adapters to access the LUN.

d. Perform a configuration update.

Define FCP Devices, SCSI Disks and Connection Paths to z/VSE

The IUI *Hardware Configuration* dialog of z/VSE is used to define the FCP devices and SCSI disks to z/VSE.

In this example (shown in Figure 31 on page 82), we define to z/VSE:

- FCP devices `FA0` and `FB0`.
- FBA-SCSI disk with cuu addresses `DA1`, and the connection path between this FBA-SCSI disk and the LUN 5600 via FCP device with cuu address `FA0` and Port-1 with WWPN `5005076300CA9A76`. Note that we cannot use any of the cuu addresses that are defined in the IOCP.
- FBA-SCSI disk with cuu addresses `DA2`, and the connection path between this FBA-SCSI disk and the LUN 5601 via FCP device with cuu address `FA0` and Port-1 with WWPN `5005076300CA9A76`.
- A second connection path between the FBA-SCSI disk with cuu address `DA1` and the LUN 5600 via FCP device with cuu address `FB0` and Port-2 with WWPN `5005076300C29A76`. This establishes a *multipathing* connection.

Finally, the definitions are processed to create the appropriate statements in the IPL procedure.

Step 1. From the main *z/VSE Function Selection* panel:

a. Select *2 Resource Definition* and press **Enter**.

b. The *Resource Definition* panel is displayed. Select *4 Hardware Configuration and IPL* and press **Enter**.

c. Select *1 Configure Hardware* and press **Enter**.

d. The *Hardware Configuration: Unit Address List* panel is displayed.

Note: You can instead use the fastpath **241** of the Interactive Interface to reach the *Hardware Configuration: Unit Address List* panel.

Step 2. Press **PF6** (ADD ADDR) and you are prompted to enter the details for the first FCP device. Key the Starting Address `FA0` (for this example) and Device Name `FCP`. Press **Enter**.

```
ADM$ADDR            HARDWARE CONFIGURATION: ADD A DEVICE

Enter the required data and press ENTER.


Specify the address or the address range of the selected device.

STARTING ADDRESS........... FA0         This may be the starting address of
                                        an address range or the only address
                                        to be added.

END ADDRESS................             This address specifies the upper
                                        limit of the address range to be
                                        added.


DEVICE NAME................ FCP         The device you want to add or a "?"
                                        to get the group selection panel.



PF1=HELP       2=REDISPLAY  3=END
```

The *Hardware Configuration: Unit Address List* panel is then displayed, showing the FCP device that has been added.

Step 3. Repeat the Step 2 to add the second FCP device `FB0`.

Step 4. Define a SCSI disk:

a. Press **PF6** (ADD ADDR) to start the procedure of defining a SCSI disk.

b. Key the Starting Address `DA1` of the first SCSI disk (in this example) and Device Name `FBA-SCSI`. Press **Enter**. The *Hardware Configuration: Disk List* is displayed, showing the example SCSI disk with cuu address `DA1`.

```
ADM$DSK2              HARDWARE CONFIGURATION: DISK LIST



Options: 2 = Alter device type code/mode                  5 = Delete a disk
         3 = Specify Shared and/or Device Down by an 'X' in the appr. column
         8 = Specify DEF SCSI command
    OPT      ADDR    DEVICE   DEVICE-TYPE  DEVICE SPEC   SHARED     DEVICE
                                 CODE          MODE                  DOWN

     _       DA1     FBA-SCSI    FBA                       _          _
     _                                                     _          _
     _                                                     _          _
     _                                                     _          _
     _                                                     _          _
     _                                                     _          _
     _                                                     _          _
     _                                                     _          _
     _                                                     _          _
     _                                                     _          _
PF1=HELP       2=REDISPLAY  3=END                      5=PROCESS
```

Step 5. Define a Connection Path.

a. Key an '**8**' (Specify DEF SCSI Command) next to the FBA-SCSI disk `DA1` and press **Enter**. The *Hardware Configuration and IPL: DEF SCSI* panel is displayed.

b. In this example, you now define the connection path between the SCSI disk with cuu address `DA1` and the FCP device with cuu address `FA0`, via the Port-1 with WWPN `5005076300CA9A76`. The SCSI disk has the LUN `5600`.

```
TAS$ICME          HARDWARE CONFIGURATION AND IPL: DEF SCSI

Enter the required data and press ENTER.




FBA ...........      DA1                 cuu of the FBA-SCSI device

FCP ...........      FA0                 cuu of the FCP device

WWPN ..........      5005076300CA9A76    World wide port name of the
                                         remote controller

LUN ...........      5600                Logical unit number of the SCSI



PF1=HELP       2=REDISPLAY  3=END
```

c. Press **Enter** and the *Hardware Configuration and IPL: DEF SCSI* panel is displayed, showing *all* the connection paths that have been created (the first entry below is from a previous entry, and is not part of this example).

```
TAS$ICMD          HARDWARE CONFIGURATION AND IPL: DEF SCSI

Enter the required data and press ENTER.

OPTIONS: 1 = ADD             2 = ALTER
         5 = DELETE

 OPT     FBA       FCP       WWPN              LUN
 _       233       C01       5005076300C693CB  5176
 _       DA1       FA0       5005076300CA9A76  5600
 _
 _
 _
 _
 _
 _
 _
 _


PF1=HELP       2=REDISPLAY  3=END                    5=PROCESS
```

d. **Press PF5 (Process) to process all the entries listed.**. The *Hardware Configuration: Disk List* panel is displayed, showing the SCSI disk you have added. Again **press PF5 (Process) to process the entries listed.** The *Hardware Configuration: Unit Address List* panel is then displayed

Step 6. Now repeat Steps 4 and 5 to:

a. Add the second SCSI disk with cuu address DA2.

b. Define the connection path between the SCSI disk with cuu address DA2 and the FCP device with cuu address FA0, via Port-1 with the WWPN 5005076300CA9A76. The SCSI disk has LUN 5601.

When you have repeated Steps 4 and 5, the *Hardware Configuration: Unit Address List* panel now includes both SCSI disks DA1 and DA2, and both FCP devices FA0 and FB0.

Step 7. Create a *multipathing* connection. Now we will create a connection path between the SCSI disk with cuu address DA1 and the FCP device with cuu address FB0, via Port-2 with the WWPN 5005076300C29A76. The SCSI disk has the LUN 5600.

a. From the *Hardware Configuration: Unit Address List* panel, key a '**3**' (Select for Further Processing) next to the FBA-SCSI disk DA1, and press **Enter**. The *Hardware Configuration: Disk List* panel is displayed, showing the SCSI disk with cuu address DA1.

b. Key an '**8**' (Specify DEF SCSI Command) next to this FBA-SCSI disk and press **Enter**. The *Hardware Configuration and IPL: DEF SCSI* panel is again displayed, showing the two connection paths you have already defined.

c. Key a '**1**' next to the connection path entry for DA1 and press **Enter**. The *Hardware Configuration and IPL: DEF SCSI* panel is displayed with the fields for FBA and LUN already completed.

```
TAS$ICME          HARDWARE CONFIGURATION AND IPL: DEF SCSI

Enter the required data and press ENTER.




FBA ...........      DA1                cuu of the FBA-SCSI device

FCP ...........                         cuu of the FCP device

WWPN ..........                         World wide port name of the
                                        remote controller

LUN ...........      5600               Logical unit number of the SCSI




PF1=HELP       2=REDISPLAY  3=END
```

d. Key the FCP cuu address `FB0` and the WWPN (for Port-2) `5005076300C29A76`, and press **Enter**. The *Hardware Configuration and IPL: DEF SCSI* panel is displayed, showing *all* the connection paths that have been created (the first entry below is not part of this example).

```
TAS$ICMD          HARDWARE CONFIGURATION AND IPL: DEF SCSI

Enter the required data and press ENTER.

OPTIONS: 1 = ADD            2 = ALTER
         5 = DELETE

 OPT     FBA       FCP       WWPN              LUN
 _       233       C01       5005076300C693CB  5176
 _       DA1       FA0       5005076300CA9A76  5600
 _       DA1       FB0       5005076300C29A76  5600
 _       DA2       FA0       5005076300CA9A76  5601
 _
 _
 _
 _
 _
 _


PF1=HELP       2=REDISPLAY  3=END                    5=PROCESS
```

e. **Press PF5 (Process) to process all the entries listed.**

f. The *Hardware Configuration: Unit Address List* panel is then displayed. **Again press PF5 (Process) to process all the entries listed.** The *Hardware Configuration: Catalog Startup Members* panel is then displayed. Press **Enter**.

Step 8. The *Job Disposition* panel is then displayed. You can select:
- Option 1. Submit the job to the VSE/POWER queue.
- Option 2. File the job in your primary ICCF library.
- Option 3. Perform both the above actions.

Press **Enter**. The following statements will now be included in the IPL procedure:

```
ADD DA1:DA2,FBA
ADD FA0,FCP
ADD FB0,FCP
DEF SCSI,FBA=DA1,FCP=FA0,WWPN=5005076300CA9A76,LUN=5600
DEF SCSI,FBA=DA1,FCP=FB0,WWPN=5005076300C29A76,LUN=5600
DEF SCSI,FBA=DA2,FCP=FA0,WWPN=5005076300CA9A76,LUN=5601
```

You can list this procedure using the LIBR LIST member.

For details of the ADD and DEF statements, refer to the *z/VSE System Control Statements*, SC33-8225.

Using JCL Statements to Define or Delete Connection Paths

You can use JCL statements to define or delete connection paths between an FBA-SCSI disk and the associated LUN. If you define a connection path, the FBA-SCSI device and the FCP device must have been added during IPL.

To define a connection path, you can enter this statement in the attention routine (AR):

```
SYSDEF SCSI,FBA=cuu,FCP=cuu2,WWPN=portname1,LUN=lun
```

Alternatively, you can define a connection path in the BG partition using this statement:

```
// SYSDEF SCSI,FBA=cuu,FCP=cuu2,WWPN=portname1,LUN=lun
```

To delete a connection path, you must:

1. Set the FBA-SCSI offline using the attention routine (AR) OFFLINE command.
2. Use this statement in the attention routine (AR):

   ```
   SYSDEF SCSI,DELETE,FBA=cuu,FCP=cuu2,WWPN=portname1,LUN=lun
   ```

Note: Any connection paths you define using the SYSDEF statement will only exist until the next IPL of your system is made. If you wish to specify *permanent* connection paths, you must use the Interactive Interface.

For details of the SYSDEF SCSI command, refer to the *z/VSE System Control Statements*, SC33-8225.

Checking Which SCSI Devices Are Available

To obtain the configuration of all SCSI devices in the system, you can use the JCL QUERY SCSI command:

```
QUERY SCSI
```

To obtain the configuration of a single SCSI device in the system, you can also use the JCL QUERY SCSI command:

```
QUERY SCSI,cuu
```

Using Multipathing to Access SCSI Disks

Multipathing means that one or more alternate connection paths exist to a SCSI disk. It is used to increase the availability of a SCSI disk.

To implement multipathing, the FCP devices used to access a SCSI disk (LUN) must be on *different* physical FCP adapters (CHPIDs). If one connection path is no longer available due to an outage of an FCP adapter, the alternate connection path is used.

An FCP card can contain more than one physical FCP adapters (CHPIDs). Because maintenance activities might affect all physical FCP adapters (CHPIDs) contained on one FCP card, you might wish to use CHPIDs belonging to *different* FCP cards in your multipathing configuration. If you also wish to protect against the possible outage of a port, you can define an alternate connection path via a different port.

The example of Figure 31 on page 82 shows a multipathing configuration for the z/VSE system in LPAR-1 and the SCSI disk device DA1. In this configuration, you are protected against outages of either the physical FCP adapter or the port.

```
DEF SCSI,FBA=DA1,FCP=FA0,WWPN=5005076300CA9A76,LUN=5600
DEF SCSI,FBA=DA1,FCP=FB0,WWPN=5005076300C29A76,LUN=5600
```

If you perform a QUERY SCSI command, the information displayed would look like this:

```
AR 0015 FBA-CUU   FCP-CUU   WORLDWIDE PORTNAME   LOGICAL UNIT NUMBER
AR 0015     DA1       FA0     5005076300CA9A76      5600000000000000
AR 0015     DA1MP     FB0     5005076300C29A76      5600000000000000
```

The first connection path that is displayed is used by z/VSE to access the LUN.

If you were to use the same WWPN, you would only be protected against an outage of the physical FCP adapter, as shown below.

```
DEF SCSI,FBA=DA1,FCP=FA0,WWPN=5005076300CA9A76,LUN=5600
DEF SCSI,FBA=DA1,FCP=FB0,WWPN=5005076300CA9A76,LUN=5600
```

Using Shared SCSI Disks

DASD sharing means that a SCSI disk can be shared between more than one z/VSE system.

DASD sharing requires that a *lock file* is defined on a shared disk. For details of how to specify a lock file for use with FCP-attached FBA-SCSI disks, see "Changing Startup When Lock File Is Stored On SCSI DASD" on page 28.

Important!

- If the lock file resides on a SCSI disk, you *must* ensure that each z/VSE system that shares this lock file has its own physical FCP adapter (CHPID).
- In each DLF command, you must specify an FCP device that references a physical FCP adapter. The FCP device used in a DLF command of another z/VSE system must reference a different physical FCP adapter.
- It is strongly recommended that you do not define other system files on the lock file device.
- If the lock file resides on a SCSI disk and the z/VSE system abends, you must ensure that you IPL your z/VSE system immediately. To do so, you must use the **same** IPL procedure that you used to IPL the z/VSE system. This is required in order to free the SCSI disk for use by other z/VSE systems.

In addition, the use of the lock file device has these restrictions:

- The lock file must not reside on the DOSRES or SYSWK1 SCSI disks.
- A multipath connection to the lock file device is not allowed (it will be rejected by z/VSE).

In Figure 31 on page 82, the lock file resides on SCSI disk DA2. This lock file is shared between ZVSE1 (the first z/VSE system) and ZVSE2 (the second z/VSE system). ZVSE1 has a connection path to the SCSI disk that uses the FCP device FA0. ZVSE2 has a connection path to the SCSI disk that uses, for example, the FCP device FB1.

The FCP device FB0 is already used by ZVSE1 for multipathing.

For this shared DASD configuration, the statements would be:

```
DEF SCSI,FBA=DA2,FCP=FA0,WWPN=5005076300CA9A76,LUN=5601
DLF UNIT=DA2,BLK=...,NBLK=...,FCP=FA0  (configured in ZVSE1)

DEF SCSI,FBA=DA2,FCP=FB1,WWPN=5005076300C29A76,LUN=5601
DLF UNIT=DA2,BLK=...,NBLK=...,FCP=FB1  (configured in ZVSE2)
```

The dialog *IPL Tailoring* supports the use of the DLF command. For details, refer to the chapter describing how to tailor IPL and system startup, in the manual *z/VSE Administration*, SC33-8224.

For details about:

- The DEF and DLF statements, refer to the manual *z/VSE System Control Statements*, SC33-8225.
- DASD sharing, refer to the manual *z/VSE Guide to System Functions*, SC33-8233.

Using the Attention Routine OFFLINE / ONLINE Commands

If you enter the command:

```
OFFLINE cuu
```

where *cuu* is the FCP device number, all connection paths containing this FCP device number will be terminated. All currently ongoing I/Os against the SCSI devices will be cancelled.

If you enter the command:

```
ONLINE cuu
```

where *cuu* is the FCP device number, all previously-defined connection paths containing this FCP device number will be reactivated.

For further details about the OFFLINE and ONLINE commands, refer to the manual *z/VSE System Control Statements*, SC33-8225.

Performing an IPL of z/VSE From a SCSI Disk

Note: The information provided here is based upon the example provided in Figure 31 on page 82. For detailed information about how to perform an IPL of z/VSE, refer to the manual *z/VSE Operation*, SC33-8239.

When you perform an IPL from a *non-SCSI* disk, the IPL process uses a zSeries channel-attachment. Until z/VSE 3.1, this was the only method you could use. From z/VSE 3.1 onwards, you can perform an IPL from an *FCP-attached device* SCSI disk. However, this is not possible using a zSeries channel-attachment.

To perform an IPL of z/VSE from an FCP-attached SCSI disk, you use the *machine loader* (a platform-independent hardware tool), and not a zSeries channel-attachment. You can start the IPL from either:
- A VM guest (described in "Initiating an IPL of z/VSE From a VM Guest").
- An LPAR (described in "Initiating an IPL of z/VSE From an LPAR" on page 93).

Prerequisites

- To perform an IPL of z/VSE from a SCSI disk, the SCSI IPL hardware feature must already be installed and enabled on your zSeries processor.
- If your z/VSE system is running under z/VM, the z/VM system must also support an IPL from SCSI.

Initiating an IPL of z/VSE From a VM Guest

This section provides an overview of the steps you must follow if z/VSE is to be IPL'd from a SCSI disk, where the IPL is initiated from a VM guest. The VM guest's virtual memory is loaded with the:
- Machine loader (a platform-independent hardware tool).
- Parameters required to access the SCSI disk, which you define using the `SET LOADDEV` command (described below).

Step 1. **Use the SET LOADDEV Command to Supply the Required Parameters**. You use the SET LOADDEV command to provide the machine loader with the parameters this program needs in order to access a SCSI disk. These are the parameters you must provide to the machine loader:
- WWPN used to access the SCSI disk.

- LUN of the SCSI disk.

Using the example shown in Figure 31 on page 82, to IPL your z/VSE system from the second SCSI disk that is accessed via the WWPN `5005076300C29A76` and has the LUN `5601000000000000`, you would use the command:

```
SET LOADDEV PORTNAME 50050763 00C29A76 LUN 56010000 00000000
```

(You must pad the LUN with zeros until it reaches 16 characters).

You can also use the QUERY LOADDEV command to display the parameters that have been set for the machine loader. In this example, if you enter Q LOADDEV, the displayed information would look like this:

```
PORTNAME  50050763 00C29A76    LUN 56010000 00000000    BOOTPROG 0
BR_LBA    00000000 00000000
```

For details of the SET LOADDEV and QUERY LOADDEV commands, refer to the manual *z/VM CP Command and Utility Reference*, SC24-6008.

Step 2. **IPL the FCP Device**.
To IPL z/VSE from a SCSI disk, you perform an IPL of the FCP device used in the connection path to the SCSI disk. The syntax of the command is:

```
IPL fcp_device_number
```

Using the example shown in Figure 31 on page 82, to IPL the FCP device with the device number (cuu address) `FA0`, you would enter:

```
IPL FA0
```

Initiating an IPL of z/VSE From an LPAR

This section provides an overview of the steps you must follow if z/VSE is to be IPL'd from a SCSI disk, where the IPL is initiated from an LPAR. You use the *Hardware Management Console* (HMC) to load the z/VSE operating system into an LPAR. For details of how to navigate to the HMC *Load* panel, you should refer to the operating procedure manual for the IBM processor you are using.

In the *Load* panel, first click **SCSI** from the selection shown. Then you must enter these values:

Load Address
This is the device number of the FCP device. In the example of Figure 31 on page 82, you would enter `0FA0` to load the FCP device with the device number (cuu address) `FA0`.

World Wide Port Name (WWPN)
This is the WWPN of the port on the ESS controller which is used to connect to the SCSI disk. In the example of Figure 31 on page 82, you would enter `5005076300C29A76`.

Logical Unit Number
The LUN number of the SCSI disk from which the z/VSE operating system is to be IPL'd. In the example of Figure 31 on page 82, you would enter `5601000000000000`.

Note: You must not change any of these fields in the *Load* panel:
- Boot program selector
- Boot record logical block address
- OS specific load parameters

(The defaults are correct).

Understanding IPL Messages Relating to SCSI Disks

During an IPL, the informational message **0I04I** displays:

- The FBA-SCSI device number (IPLDEV=...).
- SCSI parameters you have specified either using LOADDEV (under z/VM) or using the *Load* panel: FCP=..., WWPN=..., and LUN=...

Here is an example message:

```
0I04I IPLDEV=X'600',VOLSER=DOSRES,CPUID=FF0198142064
      FCP=X'D00',WWPN=5005076300CA9A76,LUN=5606000000000000
```

The FBA-SCSI device number is always the one that was used during the *previous* IPL. If the previous device number cannot be determined, z/VSE generates its own device number to be used temporarily (X'FF0' in the example below).

```
0I04I IPLDEV=X'FF0',VOLSER=DOSRES,CPUID=FF0198142064
      FCP=X'D00',WWPN=5005076300CA9A76,LUN=5606000000000000
```

z/VSE also expects a DEF SCSI command for the SYSRES SCSI disk in your IPL procedure. If the DEF SCSI commands for the SYSRES device do not specify the same parameters that were used for the IPL, they are considered to be additional connection paths. z/VSE will always first use the IPL'd path (that was defined using LOADDEV or the *Load* panel). In the example shown below, the DEF SCSI statement for the SYSRES device uses a different path than the one used for the IPL.

```
BG 0000 DEF SCSI,FBA=600,FCP=C01,WWPN=5005076300CA9A76,LUN=5606000000000000
BG 0000 DEF SCSI,FBA=601,FCP=D00,WWPN=5005076300C69A76,LUN=5607000000000000
```

If you perform a QUERY SCSI command, the information displayed would look like this:

```
AR 0015 FBA-CUU   FCP-CUU   WORLDWIDE PORTNAME  LOGICAL UNIT NUMBER
AR 0015     600       D00     5005076300CA9A76     5606000000000000    IPL path
AR 0015     600MP     C01     5005076300CA9A76     5606000000000000    DEF path
AR 0015     601       D00     5005076300C69A76     5607000000000000
```

Errors That Might Occur During Configuration

This section describes the errors that might occur when you configure your SCSI devices for use with z/VSE. Possible solutions are provided.

Error Message	Reason Code	Cause	Remedy
0S40I	0018	The FCP adapter is probably not authorized to access the port identified by *WWPN*.	Select the port (WWPN) in the *Fibre Channel Ports* selection. For details, see "Configure the SCSI Disks in the ESS Controller" on page 83.
0S40I	002F	The FCP devices used for establishing multipathing are defined for the same physical FCP adapter.	Use FCP devices defined on different physical FCP adapters. For details, see "Using Multipathing to Access SCSI Disks" on page 90.
0S40I	0023	The port identified by *WWPN* is not registered in the Nameserver of your FCP switch. Either: 1. The port does not exist (an incorrect WWPN was specified). 2. The port has been set "Offline" in the FCP switch. 3. The physical cable between FCP switch and disk controller is not properly connected.	Either: 1. Enter a valid WWPN. 2. Set the port to "Online" in the FCP switch. 3. Ensure that the physical cable between FCP switch and disk controller is properly connected.
0S40I	0102, 0018	Confirm that `FSFCMD=00000005` and that `FSFSTAT` shows `BADDEF`. In this case, the port identified by WWPN is not an Open FCP port.	Correct your settings in the *FC Ports Attributes* selection. For details, see "Configure the SCSI Disks in the ESS Controller" on page 83.
0S41I	–	Your SCSI disk does not support ANSI SCSI Version 3.	You must use a SCSI disk that supports ANSI SCSI Version 3.
0S42I	–	The block size of your SCSI disk is not **512 bytes**.	Re-configure your disk controller.
0S43I	–	The size of your SCSI disk is less than 8 MB, which is the minimum supported size.	Increase your SCSI disk size.
0S44I	–	The size of your SCSI disk is greater than 24 MB, which is the maximum supported size. z/VSE will only use the first 24 MB of your SCSI disk.	This message is provided for information only. It informs you that you are wasting disk space.
0S46I	052500	Either: 1. The LUN does not exist in the disk controller. 2. The physical FCP adapter is not authorized to access the LUN.	Either: 1. Enter a valid LUN. 2. Select the Nickname of your physical FCP adapter in the *Assign selected volumes to target hosts* selection. For details, see "Configure the SCSI Disks in the ESS Controller" on page 83.

Chapter 6. Tailoring the Interactive Interface

z/VSE provides dialogs for tailoring the Interactive Interface according to the needs of your installation. You can:

- Define user profile information.
- Change the selections offered by the Interactive Interface panels.
- Include your own CICS applications so that you can access them from the Interactive Interface.

z/VSE provides four dialogs to help you tailor the Interactive Interface according to your needs:

Maintain User Profiles
Maintain Selection Panels
Maintain Application Profiles
Maintain Synonyms

These dialogs are described in detail in this chapter. In addition, the panel *User Interface Tailoring* offers two more dialogs which allow the system administrator to:

- *Maintain PRIMARY Sublibraries*

  This function is for creating, maintaining, and deleting PRIMARY sublibraries. A VSE (not VSE/ICCF) PRIMARY sublibrary is created for any user who is authorized in the user profile.
- *Customize z/VSE Workstation Platform*

  This dialog supports workstation integration and allows the specification of up to 3 classes for file transfer between a workstation and VSE sublibraries. Refer to the manual *VSE/ESA Programming and Workstation Guide* under "The Librarian Transaction Server" for further details about this dialog.

Planning Considerations

Before you change the Interactive Interface provided by z/VSE, you should carefully plan how you want your system to look like. Review the information in the following sections about the dialogs provided.

You should be aware that:

- For each user signing on, about 2 KB of virtual storage below the 16 MB line are needed in the associated CICS partition.
- VSE/ICCF is available for type 1 and 2 users with a 4-character user ID.

  VSE/ICCF is **not** available for type 3 users and type 1 and 2 users with a 5 to 8-character user ID.

VSE CONTROL FILE

User interface tailoring is done by maintaining records in the z/VSE control file. The file is used by each CICS TS and by the BSM (Basic Security Manager). To close the file in case of an update or the alteration of VSE/VSAM options, use the following command:

```
CEMT SET FILE=(IESCNTL) CLOSE
```

Issue the command for each CICS TS. To close the control file for the BSM, issue:

```
MSG FB,DATA=CLOSECNTL
```


The control file is a VSE/VSAM KSDS file. It contains the following types of records:

1. User profile record

   A user profile record exists for each system user. The records are defined and maintained by using the *Maintain User Profiles* dialog or the batch utility IESUPDCF (described in Appendix C, "Batch Program IESUPDCF," on page 389).

   User profile records are used by the BSM for security checking.
2. Selection panel record

   Selection panel records are used to build selection panels. z/VSE ships records for all the selection panels in the Interactive Interface. You can create and maintain other records using the *Maintain Selection Panels* dialog.
3. Application profile record

   An application profile record contains execution information about a CICS application. z/VSE ships records for the Interactive Interface dialogs.

   z/VSE also provides additional application profiles which you can include in the Interactive Interface. Under "Dialogs of the Interactive Interface", the manual *z/VSE Planning* lists the dialogs and additional applications.

   You can create and maintain other application profile records using the *Maintain Application Profiles* dialog. This lets you incorporate your own CICS applications into the Interactive Interface.
4. Synonym record

   A synonym record exists for each system user who has defined synonyms for accessing dialogs. Each user can create and maintain own synonym records using the *Maintain Synonyms* dialog.
5. News record

   A news record contains a *news item* which is a message that the system displays when a user signs on. You define these records using the *Enter News* dialog.

Note: Items 6, 7, 8, and 9 are defined and maintained by the *Customize z/VSE Workstation Platform* dialog.

6. System ID record

   This record exists for each z/VSE system.
7. Class for job submission

   Defines in which class the job is to run.
8. Class for librarian transaction server

   Defines the class(es) for the execution of the transaction server.
9. Library record

   This record is for VSE libraries and sublibraries that are visible to programmable workstation users.

Maintaining User Profiles

Every user of the Interactive Interface is defined by a user profile. The profile includes a unique user ID and password which is used to sign on to the Interactive Interface. It also determines what is displayed after the user signs on. The system can display a selection panel or access an application directly. The authorization a user has to do certain tasks is also defined in the user profile.

The Interactive Interface provides the *Maintain User Profiles* dialog. You define, update, or delete user profiles using this dialog. To access the dialog, start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
1 (User Interface Tailoring)
1 (Maintain User Profiles)

Administrator Fast Path: 211	Synonym Default: upm Yours:

```
IESADMUPL2                MAINTAIN USER PROFILES
VSE CONTROL FILE
START.... ________
OPTIONS:   1 = ADD            2 = CHANGE              5 = DELETE
                      PASSWORD     REVOKE  USER INITIAL  NAME
   OPT    USERID      VALID UNTIL    DATE   TYPE  NAME    TYPE

    _     $SRV        12/01/05 *             2   IESERSUP  2
    _     AMAD                               1   IESEADM   2
    _     ANST                               1   IESEADM   2
    _     CICSUSER    12/01/05 *             3   DFLESEL   2
    _     DBDCCICS                           1   DUMEY     2
    _     ELKE        12/01/05    01/01/06   1   IESEADM   2
    _     FORSEC                             3   IESEADM   2
    _     HENZ                               1   IESEADM   2
    _     HSCZ                               1   IESEADM   2
    _     NASS                               1   IESEADM   2
    _     OPER        12/01/05 *             2   IESEOPER  2

PF1=HELP                  3=END
PF7=BACKWARD   8=FORWARD    9=PRINT
```

A FULIST displays the user IDs defined to the system. The options you can choose are shown at the top of the FULIST. Use **PF7** or **PF8** to scroll through the entries. If you want to locate a particular user ID, enter the user ID in the START field. The dialog searches for the user ID and displays it at the beginning of the list.

Creating a Status of Userids Using the Dialog

To create a status report of user profiles that are stored in the VSE Control File, press **PF9**. A status report is then created using the reporting tool IESXSPR, and stored in the VSE/POWER List Queue. The job name of this List Queue entry is IESXSUSP.

The skeleton IESXSUSP is provided in VSE/ICCF Library 59. This skeleton contains the source code of the report format. To create your own report layouts, you can modify this source code.

If you change the skeleton IESXSUSP, you must activate the related phase using the CEMT SET PROG(IESXSUSP) NEWCOPY command.

Maintaining User Profiles without VSE/ICCF

You can also update user profiles for CICS users in an environment without VSE/ICCF (in case of a second CICS, for example), or if:

- VSE/ICCF has been terminated.
- The VSE/ICCF DTSFILE has been disconnected.
- The system administrator is a non-VSE/ICCF user.

Utility Program IESUPDCF

To maintain large numbers of user profiles, you can use the batch utility program IESUPDCF which is described in detail in Appendix C, "Batch Program IESUPDCF," on page 389.

Delete a User ID

To delete a user ID, enter **5** in the OPT column to the left of the user ID. The dialog deletes the user profile record from the z/VSE control file. The VSE/ICCF DTSFILE entry for type 1 and 2 users with a 4-character user ID is deleted by a batch job created and submitted by the dialog.

If the user owns a PRIMARY sublibrary, the administrator is asked to confirm the deletion of this user ID.

Add or Update a User ID

To add a user ID, enter **1** in the OPT column next to a user ID that you want to use as a model. The model provides default values. "Additional Considerations" on page 108 has information on the default values.

To change the profile information of a user ID, enter **2** in the OPT column to the left of the user ID.

The dialog displays the *Add or Change User Profile* panel. The panel is actually one of **four** panels you use to specify profile information. The first two panels define z/VSE profile information. The third and fourth panels deal with CICS security information. After you have completed all panels, press **PF5** to update the z/VSE control file.

User Profile Information

For the first two panels, the **z/VSE profile information** is described on the following pages.

```
IESADMUPBA                ADD OR CHANGE USER PROFILE
Base     II       CICS     ResClass ICCF

To CHANGE, alter any of the entries except the userid.

  USERID............. ENDU       4 - 8 characters (4 characters for ICCF users)

  INITIAL PASSWORD... ________   3 - 8 characters

  DAYS............... 000        0-365 Number of days before password expires
  REVOKE DATE........ ________   Date when Userid will be revoked (mm/dd/yy)

  USER TYPE.......... 1          1=Administrator, 2=Programmer, 3=General
  INITIAL NAME....... IESEADM    Initial function performed at signon
  NAME TYPE.......... 2          1=Application, 2=Selection Panel
  SYNONYM MODEL...... ________   Userid to be used as model for synonyms



PF1=HELP                    3=END                    5=UPDATE
               8=FORWARD
```

Figure 33. Add or Change User Profile Panel

USERID
The user ID which identifies the user to the Interactive Interface. It can be 4 to 8 alphanumeric characters long and include the special characters @, #, or $. No blank is allowed.

Note that access to VSE/ICCF depends on the length of the user ID. A 5 - 8 character user ID does not have access to VSE/ICCF. Refer also to "Planning Considerations" on page 97.

If you are **changing** a user profile, you **cannot** change this field on the panel.

INITIAL PASSWORD
This is the password associated with the user ID. Specify 3 - 8 alphanumeric characters including the special characters @, #, or $. No blank is allowed.

The user is forced to change the password during the first sign-on.

DAYS
The number of days before the password expires. Specify a number between 0 and 365. If you enter 0, the password will **not** expire.

Information about password expiration and how you can change your password is given in "Password Expiration" on page 126.

REVOKE DATE
Enter the date, when the user ID will be revoked by the system. After this date, a sign-on attempt with this user ID will be rejected. The valid date format is MM/DD/YY. You can also specify 0 if the user ID has an unlimited validity.

Note: If the User ID is revoked you must change the revoke date, not the password.

USER TYPE
Enter one of the following:

- **1** (Administrator)

  This selection provides **VSE/ICCF** administrative authority if the user has a 4-character user ID and is defined on a CICS subsystem with VSE/ICCF. The **SYSA** profile supplied by z/VSE defines a type 1 user.

  If the user is defined for a CICS subsystem without VSE/ICCF, no VSE/ICCF administrative authority is provided.
- **2** (Programmer and Operator)

  If the user has a 4-character user ID, this selection provides access to VSE/ICCF, but not VSE/ICCF administrative authority. The **PROG** and **OPER** profiles supplied by z/VSE define type 2 users.
- **3** (General)

  This selection does **not** provide access to VSE/ICCF. It is intended for application end users. z/VSE does not supply a predefined profile for type 3 users.

INITIAL NAME
Name of the selection panel or application invoked when the user signs on to the Interactive Interface.

NAME TYPE
This defines the type of function you specify in the INITIAL NAME field.

- **1** - Application

  The system invokes the application when the user signs on.

- **2** - Selection Panel

  The system displays the selection panel when the user signs on.

SYNONYM MODEL
: This defines the user ID to be used as a model for synonyms. z/VSE provides synonyms for users SYSA, PROG, and OPER. These can be used as models for other users. For further details about synonyms refer to “Maintaining Synonyms” on page 125.

```
  IESADMUPII                    USER AUTHORIZATION
Base     II       CICS     ResClass ICCF
Answer yes or no to the following questions for userid ENDU
Enter 1 for yes, 2 for no
  NEWS...................... 1   Should user receive news items?
  ESCAPE.................... 1   Can user escape to CICS?
  CONFIRM DELETE............ 2   Does user want a confirmation message?
  VSE PRIMARY SUBLIBRARY..... 1   Does user want a PRIMARY sublibrary?
  SUBMIT TO BATCH........... 1   Can user submit to Batch?
  VSAM FILES................ 1   Can user define VSAM files?
  VSAM CATALOGS............. 1   Can user manage VSAM catalogs?
  OLPD...................... 1   Can user delete OLPD incidents?
  CONSOLE COMMANDS.......... 1   Can user enter all commands?
  CONSOLE OUTPUT............ 1   Can user see all messages?
  BATCH QUEUES.............. 1   Can user manage all POWER jobs?
  APPLICATION PROFILES....... 1   Can user maintain application profiles?
  SELECTION PANELS.......... 1   Can user maintain selection panels?
  USER PROFILES............. 1   Can user maintain user profiles?
  DEFAULT USER VSAM CATALOG.. IJSYSCT

PF1=HELP                   3=END                    5=UPDATE
PF7=BACKWARD   8=FORWARD
```

Figure 34. User Authorization Panel for Type 1 User

For the following fields, you can enter:

1 - YES
2 - NO

NEWS
: The system displays *news items* to the user.

  News items are messages which the system displays when a user signs on. It also displays the messages to users already signed on to the system. You use the *Enter News* dialog to add, change, or delete news items.

ESCAPE
: The user can *escape* to CICS. This lets the user leave the Interactive Interface and go into native CICS mode. If a user has this authorization, the selection panels show PF6 and PF9. These PF keys are used for the *escape* facility.

CONFIRM DELETE
: This defines whether the user gets a confirmation message when deleting VSE/POWER queue entries or VSE/ICCF library members.

VSE PRIMARY SUBLIBRARY
: This defines whether the user gets assigned a VSE sublibrary named PRIMARY.userid. The PRIMARY sublibrary will be created by using the *Maintain PRIMARY Sublibraries* dialog for any user who has this option set in the profile.

SUBMIT TO BATCH
This defines whether the user is authorized to submit jobs to the batch queues.

VSAM FILES
The user can define and delete VSE/VSAM files, libraries, alternate indexes, and alternate names. This authorization is **not** available for general user (type 3) profiles.

The user can access selections 1, 2, 3, and 4 from the *File and Catalog Management* dialog. These selections are:
- Display or Process a File
- Define a New File
- Define a Library
- Define an Alternate Index or Name

VSAM CATALOGS
The user can process VSE/VSAM catalogs and define and delete VSE/VSAM space. This authorization is **not** available for general user (type 3) profiles.

The user can access selections 1, 5, and 6 from the *File and Catalog Management* dialog. These selections are:
- Display or Process a File
- Display or Process a Catalog, Space
- Define a New User Catalog

OLPD The user can delete Online Problem Determination (OLPD) incident records from the system. This authorization is **not** available for general user (type 3) profiles.

CONSOLE COMMANDS
This allows the user to access a master console and to enter all commands. This is only valid for type 2 users (programmer and operator).

CONSOLE OUTPUT
If this flag is set, the user gets all messages displayed on the console. This authorization is **not** available for type 3 users.

BATCH QUEUES
A type 1 user (administrator) can manage all VSE/POWER jobs of type 1 and type 2 users. This includes displaying, changing, printing, or deleting a VSE/POWER job.

As a type 2 user, you can handle only jobs which you submitted or which are destined for you.

Note: In the dialog, you can set BATCH QUEUES to 1 (yes) also for a type 2 user if needed and assign the same authority as for a type 1 user. This requires, however, that bit 2 in the VSE/ICCF option byte OPTB (which is described later) is set to 1. You should consult the manual *VSE/ICCF Administration and Operation* for the additional authorizations given when changing the setting of bit 2 to 1.

APPLICATION PROFILES
This is only valid for administrator (type 1) profiles. It allows the user to create and maintain application profiles using the *Maintain Application Profiles* dialog.

SELECTION PANELS
This is only valid for administrator (type 1) profiles. It allows the user to create and maintain selection panels using the *Maintain Selection Panels* dialog.

USER PROFILES
This is only valid for administrator (type 1) profiles. It allows the user to create and maintain user profiles using the *Maintain User Profiles* dialog.

DEFAULT USER VSAM CATALOG
This defines the name of the user's default catalog. It is not available for type 3 users.

CICS Profile Information

For the third and fourth panel, the **CICS profile information** is described below. Refer to CICS documentation for more details on the values you can specify.

```
IESADMUPCI                 ADD OR CHANGE CICS SEGMENT
Base     II       CICS     ResClass ICCF

   OPERATOR ID............ SYA   Enter 3 character id for user ENDU
   OPERATOR PRIORITY...... 000   Operator priority between 0-255
   XRF SIGNOFF............ 2     Sign off after XRF takeover (1=yes,2=no)
   TIMEOUT................ 00    Minutes until sign off between 0-60

   PRIMARY LANGUAGE........      National language for CICS messages


     Place an 'X' next to the operator classes for this user

   01 X    02 _    03 _    04 _    05 _    06 _    07 _    08 _
   09 _    10 _    11 _    12 _    13 _    14 _    15 _    16 _
   17 _    18 _    19 _    20 _    21 _    22 _    23 _    24 _




PF1=HELP                   3=END                     5=UPDATE
PF7=BACKWARD   8=FORWARD
```

Figure 35. Add or Change CICS Profile Panel

OPERATOR ID
CICS three character operator identification. The ID should be unique.

OPERATOR PRIORITY
The value which CICS uses for the dispatching priorities of the user. Enter a number from 0 to 255.

XRF SIGNOFF

This defines if the user is signed-off after an XRF takeover. Enter 1 for YES, and 2 for NO.

TIMEOUT
Gives the value in minutes used by CICS to initiate sign off after the value specified has elapsed since the latest terminal activity. After such a timeout, you get the *z/VSE Online* panel displayed. You can specify a value from 0 to 60. The value you specify is always rounded up to a multiple of 5 minutes. A value of 0 means no time out. 0 should be specified for VSE/ICCF users.

If you specify a TIMEOUT value for a VSE/ICCF user, the CICS TIMEOUT value for this user should be greater than the sum of all ICCF TIMEOUT values. This ensures that in case of a VSE/ICCF timeout, the user affected is “reset” to a z/VSE panel (from a VSE/ICCF panel) thus enabling a correct working of a possible CICS timeout as well.

PRIMARY LANGUAGE
: Specify the language in which CICS messages should be displayed. Specify ’E’ for US English or ’J’ for Japanese. If you leave this field blank, the CICS default will be used.

OPERATOR CLASSES
: Choose the operator classes from 1 to 24. This defines the user to the CICS Transaction Server system. 1 is the default operator class.

On the fourth panel, the CICS transaction security keys and batch access rights can be specified.

```
 IESADMUPR1           ADD OR CHANGE RESOURCE ACCESS RIGHTS
Base     II       CICS     ResClass ICCF

     Place an 'X' next to the transaction security keys for user ENDU
   01 X   02 X   03 X   04 X   05 X   06 X   07 X   08 X   09 X   10 X   11 X
   12 X   13 X   14 X   15 X   16 X   17 X   18 X   19 X   20 X   21 X   22 X
   23 X   24 X   25 X   26 X   27 X   28 X   29 X   30 X   31 X   32 X   33 X
   34 X   35 X   36 X   37 X   38 X   39 X   40 X   41 X   42 X   43 X   44 X
   45 X   46 X   47 X   48 X   49 X   50 X   51 X   52 X   53 X   54 X   55 X
   56 X   57 X   58 X   59 X   60 X   61 X   62 X   63 X   64 X

     Specify the access rights for 1-32 DTSECTAB access control classes
     ( _=No access, 1=Connect, 2=Read, 3=Update, 4=Alter )
   01 _   02 _   03 _   04 _   05 _   06 _   07 _   08 _   09 _   10 _   11 _
   12 _   13 _   14 _   15 _   16 _   17 _   18 _   19 _   20 _   21 _   22 _
   23 _   24 _   25 _   26 _   27 _   28 _   29 _   30 _   31 _   32 _

   READ DIRECTORY..... 1   User can read directory with Connect (1=yes, 2=no)
   B-TRANSIENTS....... 1   User can manipulate B-Transients     (1=yes, 2=no)

PF1=HELP                    3=END                     5=UPDATE
PF7=BACKWARD   8=FORWARD
```

Figure 36. Add or Change Resource Access Rights Panel

TRANSACTION SECURITY KEYS
: You can specify from 1 to 64 keys. The default is 1. The Interactive Interface requires the use of CICS security keys 1 and 61. The security keys correspond to the TRANSEC operand in the DTSECTXN macro and the security class in the *Define Transaction Security* dialog.

ACCESS RIGHTS
: You can specify the access right for batch access classes 1 to 32. Enter 1 for connect, 2 for read, 3 for update, or 4 for alter. If nothing is specified, no access to a resource with this access class is allowed.

READ DIRECTORY
: If you specify 1, this user can read the directory of a library or sublibrary.

B-TRANSIENTS
: If you specify 1, this user can catalog, rename or delete a B-transient in a protected sublibrary if the user has the required access right to the sublibrary.

After you have entered z/VSE and CICS information on the four panels, press PF5. The dialog updates the VSE.CONTROL.FILE for the user profile. If the user is not a general (type 3) user, the dialog continues and asks for VSE/ICCF information.

General User Profiles

If you are adding or changing a general user (**type 3**) profile or profiles for **type 1 or type 2** users without access to VSE/ICCF (5 to 8-character user ID), you are finished at this point. General user profiles do not have access to VSE/ICCF, so you do not enter VSE/ICCF information. Refer also to "Planning Considerations" on page 97 for further details about the VSE/ICCF dependencies.

The dialog redisplays the FULIST of user IDs. You can process additional user IDs or press PF3 to end the dialog.

Changing or Adding a User ID

If you are changing or adding a user ID for an administrator or programmer (**type 1 or 2 user**), the dialog has updated user profile information for z/VSE and CICS in the z/VSE control file. You have now the option of updating VSE/ICCF profile information (via the **Transfer Control** panel when you are *changing* a user ID). Press the appropriate PF key:

PF5 - YES (You want to update the VSE/ICCF profile).
PF6 - NO (You do not want to update the VSE/ICCF profile).

Note that access to VSE/ICCF is only possible for users with a 4-character user ID.

The VSE/ICCF default values should be acceptable for most users. You should not change the default values unless you have a specific reason to do so. See also "Additional Considerations" on page 108.

If you press PF6 (NO), the update process is complete. The dialog redisplays the FULIST of user IDs.

For the remaining panels, the **VSE/ICCF profile information** is described below.

The dialog displays the **Specify Library** panel. In the LIBRARY field, enter the library number for the user's VSE/ICCF primary library. For further information about VSE/ICCF libraries refer to "VSE/ICCF Library Considerations" on page 109. You can accept the remaining z/VSE defaults for VSE/ICCF information or change the defaults. In the DEFAULTS field, enter one of the following:

1 - YES (You do accept the defaults.)
2 - NO (You do not accept the defaults.)

If you enter 2 (NO), you are asked for additional VSE/ICCF information. In general, the default values should be acceptable for most users. You should carefully consider any VSE/ICCF changes that you make and use the recommended values. This is to ensure that the Interactive Interface operates correctly. For more detailed information on VSE/ICCF options, refer to the manual *VSE/ICCF Administration and Operation*.

- VSE/ICCF option bytes: OPTA, OPTB, and OPTC

  The default option byte settings depend on the z/VSE user type. For administrator (type 1) profiles, the defaults are:
  OPTA - 01110001
  OPTB - 11111010
  OPTC - 01000000

  For programmer and operator (type 2) profiles, the defaults are:
  OPTA - 00000100
  OPTB - 10000000
  OPTC - 01000000

  The default settings are usually satisfactory for most users.

  In the OPTA, OPTB, and OPTC bytes, you can change certain bits identified by an asterisk (*) below.

  Note: As a general rule, you should only change bits identified by an *. If you change any other bits, the Interactive Interface may not work correctly for that user. An exception is the following situation:

  If you decide to have several type 1 users sharing one common VSE/ICCF library, you should set bit 5 of the OPTA byte to ensure that all functions of the Interactive Interface work correctly.

  – User Type 1 (Administrator)
    OPTA - 011*00*1 (You can only change bits 3,6)
    OPTB - **111010 (You can only change bits 0,1)
    OPTC - **000*0* (You can only change bits 0,1,5,7)

  – User Type 2 (Operator or Programmer)
    OPTA - 000*01*0 (You can only change bits 3,6)
    OPTB - ***00000 (You can only change bits 0,1,2)
    OPTC - **000*0* (You can only change bits 0,1,5,7)

- **VSE/ICCF security keys:**
  1 to 32 keys.
- **Alternate VSE/ICCF libraries:**
  Enter up to eight additional private VSE/ICCF libraries that the user can access (in addition to the primary library and public libraries).
- **CLASS:** Specify the default interactive partition (alphabetic).
- **MAXSTATE:** The value must be between 500 and 9999.
- **MAXPRINT:** The value cannot be greater than 9999.
- **MAXPUNCH:**
  The value cannot be greater than 32,767.
- **LINESIZE:** A value from 1-80.
- **TIMELIM:** The value cannot be greater than 32,767.
- **TIMEMAXEX:**
  The value cannot be greater than 65,535.
- **DEL:** You should **not** change the default.
- **TAB:** You should **not** change the default.
- **BS:** You should **not** change the default.
- **ESC:** You should **not** change the default.

- **END:** You should **not** change the default.
- **HEX:** You should **not** change the default.
- **LOGONRTN:**
  You should **not** change the default.
- **TIMEOUT:** You should **not** change the default.

If there is a need to change the above settings in full screen editor mode of VSE/ICCF, for example the HEX or TAB option, you must use the SET command of VSE/ICCF. If you have completed editing, you should reset the changed values to their defaults to ensure that afterwards all functions work correctly again.

Additional Considerations

Creating a Status Report of User IDs Using IESBLDUP

To maintain user profiles, you need an up-to-date record of the users defined to your system. You can print such a status report with the migration utility program (IESBLDUP), as described in “Maintaining User Profiles” on page 98.

You can also create a status report using the PRINT function of the *Maintain User Profiles* dialog. For details, see “Creating a Status of Userids Using the Dialog” on page 99.

Also refer to the manual *z/VSE System Utilities* under “IESBLDUP Utility” for a general description of the program and for a job stream example.

Dialog Considerations

1. If you change a user profile which is currently being used on the system, any new options do **not** immediately take effect. The user must sign off and sign on again to take advantage of new or changed options.
2. When you add a new user, the user ID that you enter the option number next to is used as a model. The values defined for the model are used as defaults for the new profile you are defining.

   With this, you can add new profiles using existing profiles as models. z/VSE provides user profiles for type 1 and type 2 users. If you do not need to change the defaults, you simply have to enter a new user ID and password.

   If some z/VSE defaults are not satisfactory, define a new profile and enter your own values. You can then use the new profile as a model to define other users on the system.
3. Observe the following when you add or change a profile:
   - When you add a user, select a model profile which has the same profile type (1, 2, or 3) and the same length of the user ID (4 characters or 5 to 8 characters) that you want for the new user.
   - If you change a type 1 profile to a type 2 or 3, the options that do not apply to the new user type (2 or 3) are set to 0.
   - If you add or change a profile and you change the user type, you **must update** the VSE/ICCF information. On the *Specify Library* panel, specify 2 (NO) indicating that you do not accept the defaults.

   Notes:

   a. Refer to “Planning Considerations” on page 97 for details about VSE/ICCF dependencies.

   b. If you do not update the VSE/ICCF information, the defaults for the original user profile types 1 and 2 are used as defaults for the new user ID. This could result in incorrect authorization values for the new user.
4. If you delete a type 1 or 2 user profile, you must disconnect the VSE/ICCF DTSFILE. A message requests you to do so when the job stream created is being processed.
5. z/VSE ensures that only one user can access the *Maintain User Profiles* dialog at one time.
6. Any VSE/ICCF information you enter is saved until you leave the dialog. When you finish the dialog, it makes all updates to the VSE/ICCF DTSFILE at the same time. Because of this, you can maintain several user profiles at once without waiting for the system to make the DTSFILE updates one at a time. However, when you leave the dialog, you may notice a delay while the dialog updates the DTSFILE.

VSE/ICCF Library Considerations

You can allocate VSE/ICCF libraries 3 to 7, 11 to 49, and 70 to 199 as user libraries. Other libraries are for use by the system. Further planning details about VSE/ICCF libraries are provided in the *z/VSE Planning* manual under "VSE/ICCF Libraries". Refer also to "Reformatting the VSE/ICCF DTSFILE" on page 318.

Programmer (type 2) profiles cannot access library 1. You should not define their VSE/ICCF primary library as library 1. By default bit setting, they have **read access** only to public libraries 50 - 69. However, this is only true if a user is working with the Interactive Interface. When using the command mode of VSE/ICCF, a user can switch to library 51 and read from as well as write to that library.

Note also that a FULIST for type 2 users does not display members that have been defined as shared through the VSE/ICCF utility DTSUTIL. Such members are only accessible for the owner and the system administrator.

VSE/ICCF Interactive Partitions

The manual *z/VSE Planning* has information about the characteristics and layout of the predefined VSE/ICCF interactive partitions under "VSE/ICCF Interactive Partition Layout and Characteristics". VSE/ICCF interactive partition requirements and eligibility for concurrent execution are important considerations when you create your own panel hierarchy. If you increase the size of existing interactive partitions or add class A and B partitions, you also should make a corresponding increase to the size of the CICS/ICCF (F2) partition. Interactive partitions reside in the partition GETVIS area of the CICS/ICCF partition.

VSE/ICCF DTSFILE Considerations

When the system updates the DTSFILE in an interactive partition, it uses the VSE/ICCF utility program DTSUTIL. Output from the job is put in VSE/ICCF library member U$xxxx.P (xxxx is your user ID). The member is in your default primary library. The system replaces the contents of U$xxxx.P each time you run this task.

If there is a power failure or other system interruptions before the dialog ends and updates to the DTSFILE are complete, the contents of the z/VSE control file and the DTSFILE may not match. If you think that this has occurred, do the following:
1. Access the *Maintain User Profiles* dialog again.
2. Select the CHANGE option (2) for the user profile(s) you were working with.

3. Request an update (PF5) of the VSE/ICCF information. This is necessary to ensure that all profile information is consistent.

If you do these steps, the information in the DTSFILE and the z/VSE control file will agree with each other.

Maintaining Selection Panels

z/VSE lets you change the structure of the Interactive Interface. You can create your own selection panels and corresponding HELP panels. In this way, you can have many interactive panel hierarchies for different users of your system.

Maintaining Selection Panels without VSE/ICCF

You can also update selection panels in an environment without VSE/ICCF (in case of a second CICS, for example), or if:

- VSE/ICCF has been terminated.
- The VSE/ICCF DTSFILE has been disconnected.
- The system administrator is a non-VSE/ICCF user.

Introduction

The *Maintain Selection Panels* dialog helps you create, change, or delete selection panels. You define the selections you want on the panel and specify what is invoked for each selection. Each selection can invoke:

1. Another selection panel. It can be a panel shipped by z/VSE or one that you create.
2. An Interactive Interface dialog.
3. An additional z/VSE application.

   z/VSE provides a number of applications that are not included in the default panel hierarchies of the Interactive Interface. If you wish, you can invoke one or more of these applications from selection panels that you create.

   Under “Additional z/VSE Applications”, the manual *z/VSE Planning* has an appendix that gives an overview of those applications not included in the default panel hierarchies.
4. One of your own CICS applications.

   You must define your application to the system using the *Maintain Application Profiles* dialog.

Each selection panel is defined by a selection panel record. The system stores the records in the z/VSE control file.

You can also write your own HELP information for the selection panels you create. You use the *Maintain Selection Panels* dialog to process the HELP text. The system stores HELP text in the system’s text repository file IESTRFL. You can use the dialog to add, update, or delete HELP information in the text file.

z/VSE automatically manages the display of HELP panels that you create. It displays your HELP text whenever you press PF1 from a selection panel that you have created. z/VSE automatically handles backward and forward scrolling. “Creating HELP Panels” on page 114 describes how you create HELP panels.

Before you create selection panels, refer to *z/VSE Planning* under "Planning for Tailoring the Interactive Interface" for additional information about user interface tailoring. To access the dialog, start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
1 (User Interface Tailoring)
2 (Maintain Selection Panels)

Administrator Fast Path: 212	Synonym Default: spm Yours:

An entry panel of the *Maintain Selection Panels* dialog appears where you can press ENTER to list all available panel names, or enter the name or the first characters of the panel you want to be listed.

```
IESADMUIFS           MAINTAIN SELECTION PANELS

Specify the prefix of the Selection Panels you want to be listed and
press the ENTER key.



  SELECTION.......... ________    1 - 8 prefix characters, e.g.
                                  'AB' for all Selection Panels
                                  starting with AB.
                                  Press ENTER to list all Selection
                                  Panels.







PF1=HELP                    3=END       4=RETURN
```

A FULIST displays the selection panels defined for the system. If you want to locate a particular entry, enter the selection panel name in the LOCATE NAME field.

```
IESADMSPL                         MAINTAIN SELECTION PANELS     Page   1 of  10

CONTROL FILE
OPTIONS:   1 = ADD            2 = CHANGE             5 = DELETE
           6 = UPDATE HELP     7 = DELETE HELP

 OPT     PANEL NAME  HELP  SELECTION   SELECTABLE PANELS OR APPLICATIONS

  _       IESEADM     *       1-4     IESEINST IESEDEF  IESEOPS  IESEPROB
                              5-8     IESEGDEV IESNICCF IESECICA
                                9
  _       IESEASAV    *       1-4     IESEVSAM IESELIBR IESS$BAC IESS$RHS
                              5-8     IESEBKDT IESERSTD IESECPDD
                                9
  _       IESEBKCA    *       1-4     IESC$BMT IESC$BMD IESC$BUT IESC$BUD
                              5-8
                                9
  _       IESEBKDT    *       1-4     IESU$DMV IESU$DMF
                              5-8
                                9
PF1=HELP       2=REFRESH    3=END       4=RETURN
               8=FORWARD    9=PRINT

LOCATE NAME ==> ________
```

The options you can choose are shown at the top of the FULIST. Enter an option number in the OPT column to the left of the panel you want to process.

The dialog processes HELP text whenever you select options:
- 1 (ADD)
- 6 (UPDATE HELP)
- 7 (DELETE HELP)

When you select one of these options, the dialog searches the following libraries for the VSE/ICCF library member *which has the same name as your selection panel*:
- Primary library
- Connected library
- Common library

It then either copies the member to the text repository file or deletes it from the text file, depending on the option you selected.

Selection panel names that begin with the following characters are reserved:
- IES
- INW

You **cannot** change or delete them. You can use them as models to define your own panels.

To create a status report of selection profiles/panels that are stored in the VSE Control File, press **PF9**. A status report is then created using the reporting tool IESXSPR, and stored in the VSE/POWER List Queue. This List Queue entry has the job name IESXSTX.

The skeleton IESXSTX is provided in VSE/ICCF Library 59. This skeleton contains the source code of the report format. To create your own report layouts, you can modify this source code.

If you change the skeleton IESXSTX, you must activate the related phase using the CEMT SET PROG(IESXSTX) NEWCOPY command.

Add or Change a Panel

If you add a new panel, enter option number **1** next to the panel you want to use as a model. The model provides default values.

If you change a panel, enter option number **2** next to the panel you want to change.

After you make your selection, the dialog displays an additional panel. You need the following information:

SELECTION PANEL NAME
: Specify a unique name (when adding only) for the selection panel. The name cannot begin with the characters IES or INW. These prefixes are reserved for z/VSE.

SEQ
: The sequence numbers of the selections on the panel. You can specify the numbers **1 - 9**, for up to nine options on the panel. The dialog automatically sorts the sequence numbers and the corresponding selection text in ascending order.

NAME

Enter a **1 - 8** character name indicating what is invoked when this selection is chosen. It can be:

- An application profile name.

  It can be a z/VSE dialog or application or your own CICS application which you have added using the *Maintain Application Profiles* dialog.
- The name of another selection panel.

  It can be a panel shipped with the system or one that you create.

Under “Additional z/VSE Applications”, the manual *z/VSE Planning* lists the dialogs and additional application profiles which z/VSE provides.

TYPE This indicates whether you entered an application profile or a selection panel name in the NAME field. Enter:
1 - Application profile
2 - Selection panel

SELECTION TEXT

This is the explanation text that is shown to the right of the sequence number on the selection panel.

After you type in your information, press **ENTER**. The dialog formats the information, checks for editing errors, and redisplays the panel. Check your entries and make any changes.

When you are done, press **PF5** to update the z/VSE control file and store the selection panel record. If you are adding a new panel, the dialog also searches for corresponding HELP text. If it locates the VSE/ICCF library member, it formats the HELP text and adds it to the text repository file. If you are changing a selection panel, the dialog does **not** process HELP text.

The dialog continues and redisplays the FULIST.

Delete a Panel

Option 5 (DELETE) deletes an existing selection panel record from the z/VSE control file. If you have HELP text for the panel, the dialog also deletes it from the text repository file. However, it does **not** delete the library member which contains the HELP text from the VSE/ICCF library.

Use option 7 to delete your HELP text in both the system’s text repository file and the VSE/ICCF library member.

Update HELP

Option 6 (UPDATE HELP) replaces the selection panel HELP text in the system text file with HELP text from the VSE/ICCF library member.

Delete HELP

Option 7 (DELETE HELP) deletes the selection panel HELP text from both the system text file and the VSE/ICCF library member that contains the HELP information.

The dialog does not check whether the VSE/ICCF member is found. If the correct library is not accessed, the member may not be deleted.

Rebuild Default Selection Panels

z/VSE ships selection panels for three default hierarchies:
- System administrator
- Programmer
- Operator

"z/VSE Profiles" on page 1 describes the default hierarchies.

If the default selection panel records are damaged, you can rebuild them. This can only be done using the default administrator user ID SYSA.

When user ID SYSA accesses the dialog, the FULIST displays PF6=SYSTEM. PF6 is only displayed for user ID SYSA. It rebuilds the shipped selection panel records for the three default hierarchies.

Creating HELP Panels

You can create your own HELP panels for the selection panels you create. You simply create a VSE/ICCF library member with the same name as the name of your selection panel. For example, if you create a selection panel named USERSEL, create a VSE/ICCF library member named USERSEL for your HELP text.

You can use the *Program Development Library* dialog to create VSE/ICCF library members. Under "Handling VSE/ICCF Library Members", the manual *VSE/ESA Programming and Workstation Guide* describes the dialog in detail.

After you create your library member, edit the member and type in your HELP text.

Do not enter lines longer than 68 characters. Lines which are longer are truncated. You can have blank lines, but trailing blanks are suppressed.

The system formats the HELP text in a way that one panel (page) of HELP text consists of sixteen lines, including blank lines. The text can have a maximum of 4000 characters, not including trailing blanks. This is approximately 6 - 8 panels of text. Note that you do not need to define how the system should manage the panel display of HELP text or forward and backward paging. The system does this automatically for you.

After creating the HELP text, you can incorporate it into the system using the *Maintain Selection Panels* dialog.

Additional Considerations

1. z/VSE ensures that only one user can access the *Maintain Selection Panels* dialog at one time.
2. Do not use the following prefixes for the name of your selection panels:
   - IES
   - INW

   These prefixes are reserved by z/VSE.
3. If you create HELP text before you create your selection panel, the dialog automatically adds the HELP to the system text file when you add (option 1) the new selection panel.

   If you create the HELP text after you create your selection panel, you can add the HELP information to the text file using option 6 (UPDATE HELP).

4. When you use the following options, the correct VSE/ICCF libraries must be accessed for correct HELP text processing:
   - 1 (ADD)
   - 6 (UPDATE HELP)
   - 7 (DELETE HELP)

   When the dialog searches for the VSE/ICCF library member with the same name as the selection panel, it searches in the following order:
   - Primary library
   - Connected library
   - Common library
5. If you specify that one of your own CICS applications is invoked from a selection panel, the application must be defined by an application profile record. You can define your own applications using the *Maintain Application Profiles* dialog.

Maintaining Application Profiles

You can include your own CICS applications in the z/VSE system and access them from the Interactive Interface. The application can be accessed from a selection panel or invoked directly when a user signs on.

Maintaining Application Profiles without VSE/ICCF

You can also update application profiles in an environment without VSE/ICCF (in case of a second CICS, for example), or if:

- VSE/ICCF has been terminated.
- The VSE/ICCF DTSFILE has been disconnected.
- The system administrator is a non-VSE/ICCF user.

The *Maintain Application Profiles* dialog helps you include your own applications in the Interactive Interface. Each application is defined by an *application profile record*. The record defines the name and characteristics of the application. The system stores application profile records in the z/VSE control file.

Before you include an application, review *z/VSE Planning* under "Planning for Tailoring the Interactive Interface" for information on user interface tailoring. There are many things to consider in terms of user profiles and selection panels before you change your system.

To access the dialog, start with the *z/VSE Function Selection* panel and select:
2 (Resource Definition)
1 (User Interface Tailoring)
3 (Maintain Application Profiles)

```
Administrator
Fast Path:  213
```

```
Synonym
Default:     apm   Yours:
```

An entry panel of the *Maintain Application Profiles* dialog appears where you can press ENTER to list all available application profiles, or enter the name or the first characters of the profile you want to be listed.

```
IESADMUIFA           MAINTAIN APPLICATION PROFILES

Specify the prefix of the Application Profiles you want to be listed
and press the ENTER key.



  APPLICATION........ ________   1 - 8 prefix characters, e.g.
                                 'AB' for all Application Profiles
                                 starting with AB.
                                 Press ENTER to list all Application
                                 Profiles.









PF1=HELP                   3=END        4=RETURN
```

A FULIST displays the applications defined for the system. If you want to locate a particular entry, enter the application name in the LOCATE NAME field.

```
IESADMAPL                      MAINTAIN APPLICATION PROFILES    Page   1 of  14

CONTROL FILE
OPTIONS:   1 = ADD             2 = CHANGE             5 = DELETE

                                    EXECUTION   UPPER     SHOW    SYSTEM USE
 OPT       NAME        ACTIVATE       CODE      CASE      INPUT      ONLY

  _      IESA$FST      DTRDDMGR        6         1         1         1
  _      IESA$HDW      DTRDDMGR        6         1         1         1
  _      IESA$LB       DTRDDMGR        6         1         1         1
  _      IESA$NLS      DTRDDMGR        6         1         1         1
  _      IESAPM        IETF            1         1         1         0
  _      IESAPM2       IESA            1         1         1         0
  _      IESBQU        IESQ            1         1         1         0
  _      IESC$ACT      DTRDDMGR        6         1         1         1
  _      IESC$APP      DTRDDMGR        6         1         1         1
  _      IESC$BMD      DTRDDMGR        6         1         1         1
  _      IESC$BMT      DTRDDMGR        6         1         1         1

PF1=HELP       2=REFRESH    3=END        4=RETURN
               8=FORWARD    9=PRINT

LOCATE NAME ==> ________
```

The options you can choose are shown at the top of the FULIST. Enter the appropriate option number in the OPT column to the left of the application you want to process.

To create a status report of application profiles that are stored in the VSE Control File, press **PF9**. A status report is then created using the reporting tool IESXSPR, and stored in the VSE/POWER List Queue. The List Queue entry is called IESXSAP.

The skeleton IESXSAP is provided in VSE/ICCF Library 59. This skeleton contains the source code of the report format. To create your own report layouts, you can modify this source code.

If you change the skeleton IESXSAP, you must activate the related phase using the CEMT SET PROG(IESXSAP) NEWCOPY command.

Here is an example of the use of IESXSAP to print application profiles to the VSE/POWER List Queue.

```
 VSE APPLICATION PROFILE LIST
 VSE CONTROL FILE
         PROFILE         EXECUTION
 NUMBER  NAME     ACTIVATE CODE   PARAMETER (30 CHAR)
     1  IESA$FST DTRDDMGR  6     $$$$EASY ADM$FST

     2  IESA$HDW DTRDDMGR  6     EZSIZE=400.$$$$EASY ADM$HDW

     3  IESA$LB  DTRDDMGR  6     $$$$EASY ADM$LB

     4  IESA$NLS DTRDDMGR  6     $$$$EASY TES$MSG  27 482

     5  IESAPM   IETF      1     APM

     6  ........
```

System Provided Application Profiles

z/VSE provides a number of application profiles. The names of these profiles begin with the prefix IES and are reserved by the system. You **cannot** change or delete them. You can use them as models when you add your own application. Useful applications are, for example:

```
IESDITTO    Access DITTO/ESA for VSE
IESISQL     Access DB2 Server for VSE
```

Add or Change an Application Profile

If you add a new application profile, enter option number **1** next to the profile you want to use as a model. The model provides default values. If you change a profile, enter option number **2** next to the profile you want to change.

After you make your selection, the dialog displays an additional panel. You need the following information:

NAME
: Required when adding an application profile. Specify a unique application name of **1 - 8** characters. It identifies the application to the system.

CODE
: Specify how the application is initiated.

- **1** = Initiate transaction via CICS START.
- **2** = LINK to a program. A CICS LINK is performed to a CICS program using the current TCA.
- **3** = ATTACH a non-conversational transaction. The transaction begins as if a transaction code had been entered from the terminal. For non-conversational transactions, END-OF-TASK does not necessarily mean “end of application”. You have two choices:
  1. You can add a line of code to your last transaction program so that it transfers control back to the Interactive Interface. (See Figure 37 on page 120.)
  2. The user can press **PF3** to return to the Interactive Interface.
- **4** = ATTACH a conversational transaction. The transaction begins as if a transaction code had been entered from the terminal. For conversational transactions, z/VSE assumes that END-OF-TASK means “end of the application”. The user is automatically returned to the selection panel.

It is recommended that you use the CICS START technique (CODE=1) for not directly connected applications.

You can only use one of these four codes. Some z/VSE application profiles use different codes besides the four listed above. If you select a z/VSE profile as a model and it is defined with another code, the dialog sets the CODE field to an underscore (_) when it displays the panel. Enter a code (**1 - 4**) for your own applications.

Refer also to "Example of Application Coding for the Interactive Interface" on page 119.

ACTIVATE
Specify the name to activate the application.

If you enter 2 for CODE, this is a **1 - 8** character program name. If you enter 1, 3, or 4 for CODE, this is a **1 - 4** character transaction ID.

CASE How terminal input is passed to the application:
- **1** = Uppercase (CICS performs Upper Case Translation – UCTRAN).
- **2** = Upper- and lowercase (CICS does not perform UCTRAN).

DATA Up to 136 characters of data which is passed to the transaction or program.

If CODE=1, data is passed as interval control data. For the other three codes, data is passed in the TIOA. Note that "Example of Application Coding for the Interactive Interface" on page 119 has coding examples for retrieving data.

SHOW
Used only if you specify input data (DATA). Specify whether the data which is passed should be displayed on the user's terminal:
- **1** = data displayed on user's terminal.
- **2** = data not displayed on user's terminal.

After you type in your information, press **ENTER**. The dialog formats the information and redisplays the panel. Check your entries and make any changes.

When you are done, press **PF5** to update the z/VSE control file and store the information. The dialog continues and redisplays the FULIST.

Delete an Application Profile

Option 5 deletes an existing application profile record from the z/VSE control file. You cannot delete applications with the prefix IES, INW, and INF.

Rebuild Default Application Profiles

z/VSE ships application profile records for each Interactive Interface dialog. If any application profiles for these dialogs are damaged, you can rebuild them. This can only be done using the default administrator user ID SYSA. When user ID SYSA accesses the dialog, the FULIST displays PF6=SYSTEM. PF6 is only displayed for user ID SYSA. It rebuilds the application profile records which z/VSE provides for the Interactive Interface.

Additional Considerations

1. Do not use the prefixes IES and INW for the names of your application profiles. These prefixes are reserved for z/VSE.

2. If you add your own CICS applications, they can be invoked either from a selection panel or directly after signing on as specified in the user's profile. You can maintain selection panels and user profiles using the following dialogs:
   - *Maintain User Profiles*
   - *Maintain Selection Panels*
3. After integrating an application, you may have to modify certain CICS tables. Under "Overview on CICS Skeletons and Tables", the manual *z/VSE Planning* shows the CICS tables which z/VSE provides.

Function Selection Within an Application

Many applications present their own menus (selection panels). Some require a keyword along with the initial transaction code to access a subfunction. You can simplify how end users work with these applications by creating a selection panel that replaces the application menu. The panel's selections can be the different subfunctions of the application. In this way, you have:

- Uniform types of selection panels throughout your system.
- The ability to create your own HELPs for an application.
- A way for users to access new subfunctions. Additional selections on the panel can be used for new subfunctions.

For applications in which a single transaction code is entered with a key word for the desired subfunction, you can point to different application profile records in a selection panel record. Each application profile names the same transaction code, but passes the former key word as data. The correct application subfunction is thus presented to the user.

Example of Application Coding for the Interactive Interface

The **entry** and **exit** of an application in the Interactive Interface must be prepared differently, depending on the CODE value in the application profile record. Refer to Figure 46 on page 125 for an example of the *Maintain Application Profiles* dialog, where the CODE value is defined.

Entry and exit handling is as follows:

Code 1 or 3
- Entry – Do a normal RECEIVE of TIOA data.
- Exit – Do an XCTL to program IESFPEP.

Codes 2 or 4
- Entry – Do a normal RECEIVE of TIOA data.
- Exit – Do a normal RETURN to CICS.

Figure 37 is an example of command level coding for Code 1.

```
When you enter the application, check how it was started:
       .
       EXEC CICS HANDLE CONDITION,    Set up in case it was not
            ENDDATA(NOTVSE),          started by the Interactive
            NOTFND(NOTVSE)            Interface.

       EXEC CICS RETRIEVE,            Retrieve data passed by the
            SET(SOMEREG),             Interactive Interface.  Give
            LENGTH(HALFWORD)          register for data address and
                                      how long it is.
*
* If START with data is possible by other means, check for character
* string that could only have come from the Interactive Interface.
*
       CLC     =C'MY CHAR STRING',0(SOMEREG)
       BNE     NOTVSE
*
* Next instruction assumes program started by the Interactive Interface.
*
NOTVSE DS      0H     Come here if not started by Interactive Interface.
        .
        .
END    DS      0H     Return to Interactive Interface only if we
        .             came from there.
        .
       EXEC CICS XCTL PROGRAM('IESFPEP')
```

Figure 37. CICS Command Level Coding Example for Code 1 (Start)

Creating a User-Defined Selection Panel

This section describes the steps necessary for creating a user-defined selection panel. The example used shows how to create a panel with the following selections:

1 User Application A
2 Personal Computer Move Utilities
3 Program Development
4 File Management
5 Retrieve Message
6 Display Active Users/Send Message
7 Maintain Synonyms

Selection 1 gives access to a user-provided application. Selections 2 through 7 are standard z/VSE dialogs.

Note: The *VSE/ESA Programming and Workstation Guide* describes selections 2 through 7 in detail.

The following steps are required for creating and using the selection panel outlined above:

1. Create a user profile with the *Maintain User Profiles* dialog.
2. Create the selection panel with the *Maintain Selection Panels* dialog.
3. Create an application profile for *User Application A* with the *Maintain Application Profiles* dialog.

For the example, the following main parameters are used:

User ID:	ENDU
User Type:	2 (Programmer)
Password:	PNV48

Initial Name:	ENDSEL (Name of selection panel)
Catalog Name:	IJSYSCT
Primary Library:	16
Name:	USAPPL (Name of user application)
Activate:	APIS (Program or transaction to be activated)

The 4-character user ID (ENDU) allows this programmer to access and use VSE/ICCF.

Creating the User Profile

The *Maintain User Profiles* dialog is used to enter the information shown below. There are four panels for entering user profile information and one or more panels to define the user's VSE/ICCF primary library and VSE/ICCF parameters.

The values you can enter are described in detail under "Add or Update a User ID" on page 100.

First Panel

```
IESADMUPBA                ADD OR CHANGE USER PROFILE
Base     II       CICS     ResClass ICCF

To CHANGE, alter any of the entries except the userid.

  USERID............. ENDU       4 - 8 characters (4 characters for ICCF users)

  INITIAL PASSWORD... ________   3 - 8 characters

  DAYS...............  80        0-365 Number of days before password expires
  REVOKE DATE........ 12/01/01   Date when Userid will be revoked (mm/dd/yy)

  USER TYPE.......... 1          1=Administrator, 2=Programmer, 3=General
  INITIAL NAME....... ENDSEL     Initial function performed at signon
  NAME TYPE.......... 2          1=Application, 2=Selection Panel
  SYNONYM MODEL...... ________   Userid to be used as model for synonyms




PF1=HELP                    3=END                      5=UPDATE
               8=FORWARD
```

Figure 38. First Panel of Defining a User Profile

After entering the information you must press **PF8** (FORWARD) for the next panel.

Second Panel

```
IESADMUPII                    USER AUTHORIZATION
Base     II       CICS     ResClass ICCF
Answer yes or no to the following questions for userid ANST
Enter 1 for yes, 2 for no
  NEWS...................... 1   Should user receive news items?
  ESCAPE.................... 1   Can user escape to CICS?
  CONFIRM DELETE............ 2   Does user want a confirmation message?
  VSE PRIMARY SUBLIBRARY.... 1   Does user want a PRIMARY sublibrary?
  SUBMIT TO BATCH........... 1   Can user submit to Batch?
  VSAM FILES................ 1   Can user define VSAM files?
  VSAM CATALOGS............. 1   Can user manage VSAM catalogs?
  OLPD...................... 1   Can user delete OLPD incidents?
  CONSOLE COMMANDS.......... 1   Can user enter all commands?
  CONSOLE OUTPUT............ 1   Can user see all messages?
  BATCH QUEUES.............. 1   Can user manage all POWER jobs?


  DEFAULT USER VSAM CATALOG.. IJSYSCT

PF1=HELP                    3=END                    5=UPDATE
PF7=BACKWARD   8=FORWARD
```

Figure 39. Second Panel of Defining a User Profile for a Type 2 User

After entering the information you must press **PF8** (FORWARD) for the next panel.

Third Panel

Select one or more operator class values from 1 to 24, which identify this user to the CICS Transaction Server system. Class 1 is set per default if you do not specify other operator classes. For a detailed description of these characteristics consult the CICS Transaction Server documentation.

```
IESADMUPCI                ADD OR CHANGE CICS SEGMENT
Base     II       CICS     ResClass ICCF

   OPERATOR ID............ SYA   Enter 3 character id for user ENDU
   OPERATOR PRIORITY....... 000   Operator priority between 0-255
   XRF SIGNOFF............ 2     Sign off after XRF takeover (1=yes,2=no)
   TIMEOUT................ 00    Minutes until sign off between 0-60

   PRIMARY LANGUAGE........       National language for CICS messages


     Place an 'X' next to the operator classes for this user

   01 X    02 _     03 _     04 _     05 _     06 _     07 _     08 _
   09 _    10 _     11 _     12 _     13 _     14 _     15 _     16 _
   17 _    18 _     19 _     20 _     21 _     22 _     23 _     24 _



PF1=HELP                    3=END                    5=UPDATE
PF7=BACKWARD   8=FORWARD
```

Figure 40. Third Panel of Defining a User Profile for a Type 2 User

After entering the information you must press **PF8** (FORWARD) for the next panel.

Fourth Panel

Note that the transaction security keys and the access rights chosen are examples. When planning security for your installation consult the CICS Transaction Server documentation.

```
IESADMUPR1          ADD OR CHANGE RESOURCE ACCESS RIGHTS
Base     II       CICS     ResClass ICCF

     Place an 'X' next to the transaction security keys for user ENDU
   01 X   02 X   03 X   04 X   05 X   06 X   07 X   08 X   09 X   10 X   11 X
   12 X   13 X   14 X   15 X   16 X   17 X   18 X   19 X   20 X   21 X   22 X
   23 X   24 X   25 X   26 X   27 X   28 X   29 X   30 X   31 X   32 X   33 X
   34 X   35 X   36 X   37 X   38 X   39 X   40 X   41 X   42 X   43 X   44 X
   45 X   46 X   47 X   48 X   49 X   50 X   51 X   52 X   53 X   54 X   55 X
   56 X   57 X   58 X   59 X   60 X   61 X   62 X   63 X   64 X

     Specify the access rights for 1-32 DTSECTAB access control classes
     ( _=No access, 1=Connect, 2=Read, 3=Update, 4=Alter )
   01 _   02 _   03 _   04 _   05 _   06 _   07 _   08 _   09 _   10 _   11 _
   12 _   13 _   14 _   15 _   16 _   17 _   18 _   19 _   20 _   21 _   22 _
   23 _   24 _   25 _   26 _   27 _   28 _   29 _   30 _   31 _   32 _

   READ DIRECTORY..... 1   User can read directory with Connect (1=yes, 2=no)
   B-TRANSIENTS....... 1   User can manipulate B-Transients     (1=yes, 2=no)

PF1=HELP                    3=END                    5=UPDATE
PF7=BACKWARD   8=FORWARD
```

Figure 41. Fourth Panel of Defining a User Profile

After entering the information you must press **PF5** (UPDATE) for processing the data you entered in the last four panels.

Panel for Specifying VSE/ICCF Primary Library

```
                                                 User Id = ENDU


LIBRARY.................... 16     The primary library for this user.



DEFAULTS................... 1      Do you accept the remaining defaults?
                                   Enter 2 = no, 1 = yes.
                                   (Do not change defaults, without care-
                                    ful consideration)
```

Figure 42. Panel for Defining Primary Library in User Profile

After entering the information you must press **ENTER** for processing.

Creating the Selection Panel

The *Maintain Selection Panels* dialog is used to enter the information shown below. In the first panel, IESEADM is chosen as a model for the selection panel. In the second panel, the required selections are defined.

Note: For the *name* to be specified for an application refer also to the manual *z/VSE Planning*. In this manual, the appendixes “Dialogs of the Interactive Interface” and “Additional z/VSE Applications” show the names of the application profiles provided and which can be specified for selection panels.

```
OPT    PANEL NAME   HELP   SELECTION   SELECTABLE PANELS OR APPLICATIONS

 1      IESEADM      *        1-4    IESEINST IESEDEF  IESEOPS  IESEPROB
                              5-8    IESEGDEV IESNICCF IESECICA
                                9
 _      IESEASAV     *        1-4    IESEVSAM IESELIBR IESS$BAC IESS$RHS
                              5-8    IESEBKDT IESERSTD IESECPDD
                                9
          .
          .
          .
```

Figure 43. First Panel of Defining a Selection Panel

After entering the information you must press **ENTER**.

```
SELECTION PANEL NAME.... ENDSEL

SEQ  NAME    TYPE             SELECTION TEXT

 1  USAPPL     1  User Application A
 2  IESEIWS    2  Personal Computer Move Utilities
 3  IESEGDEV   2  Program Development
 4  IESVSAM    1  File Management
 5  IESIMSG    1  Retrieve Message
 6  IESUSER    1  Display Active User/Send Message
 7  IESSYN     1  Maintain Synonym
 8  IESISQL    1  Access DB2 Server for VSE
 9             1
```

Figure 44. Second Panel of Defining a Selection Panel

After entering the information you must press **PF5** (UPDATE) for processing.

Creating the Application Profile

The *Maintain Application Profiles* dialog is used to enter the information shown below. In the first panel, IESAPM is chosen as a model for the application profile. In the second panel, the application related information is entered.

```
       APPLICATION                EXECUTION  UPPER    SHOW   SYSTEM USE
 OPT       NAME       ACTIVATE      CODE     CASE     INPUT     ONLY

  _       IESA$FST      DTRDDMGR       6        1        1        1
  _       IESA$HDW      DTRDDMGR       6        1        1        1
  _       IESA$LB       DTRDDMGR       6        1        1        1
  1       IESAPM        IESA           1        1        1        0
            .
            .
            .
```

Figure 45. First Panel of Defining an Application Profile

After entering the information you must press **ENTER**.

```
NAME......... USAPPL   Unique application name, 1-8 characters.

CODE......... 1        1=START trans ID, 2=LINK to program, 3=ATTACH NON-
                       CONVERSATIONAL trans ID with data, 4=ATTACH
                       CONVERSATIONAL trans ID with data.

ACTIVATE..... APIS     Name to activate, a 1-8 character program name or
                       a 1-4 character transaction ID.

CASE......... 1        Terminal input passed to application in uppercase
                       only(CASE=1) or upper/lowercase(CASE=2).

DATA......... USERDATA

                                                                     <==

                       Optional input data to pass to application.

SHOW......... 2        Show input data(SHOW=1) or do not show it(SHOW=2).
```

Figure 46. Second Panel of Defining an Application Profile

After entering the information you must press **PF5** (UPDATE) for processing.

Accessing the Newly Created Selection Panel

After completing the above steps, user ENDU can log on with password PNV48. The *z/VSE Function Selection Panel* displayed shows the selections defined for user ENDU. User ENDU can work with the selections displayed except for selection 1 (*User Application A*). To access *User Application A*, the application must first be installed (define programs, maps, transactions, and so on to CICS). For an overview on how to install a user-written application, refer to the manual *z/VSE Planning* under “Adding a User-Written Application to Your System”.

Maintaining Synonyms

With this function you can define private synonyms for accessing panels. To access the dialog, start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
1 (User Interface Tailoring)
4 (Maintain Synonyms)

Administrator Fast Path: 214	Synonym Default: synonyms Yours:

When selecting this dialog, you get a list of synonyms active for your user ID. You can locate a particular entry by using the LOCATE field.

Adding, Changing, or Deleting a Synonym

To **add** a synonym, press **PF6**. You get the *Add New Synonyms* panel. On this panel you can enter up to 13 new synonyms and paths. You can view the updated list of synonyms by pressing PF6 again. In addition, you can **change** a synonym (option 2) and **delete** a synonym (option 5).

A synonym must consist of 1-8 alphameric characters, including the characters $, #, and @. The first character cannot be a number.

Additional Considerations

Users can have their own private synonyms or use a synonym model defined in their user profile by the system administrator. z/VSE provides synonyms for users SYSA, PROG, and OPER. They can be used as models for other user IDs. Appendix A, "Fast Paths and Synonyms for Dialogs," on page 377 shows the synonyms for these users.

You assign a synonym model by specifying the user ID of the synonym model owner in the SYNONYM MODEL parameter of a user's profile. Specifying PROG, for example, allows a user to use the synonyms defined for user ID PROG. You define the SYNONYM MODEL parameter with the *Maintain User Profiles* dialog. Refer to "Maintaining User Profiles" on page 98 for a description of the dialog and the SYNONYM MODEL parameter. To change synonyms, the following applies:

- Private synonyms can be changed by each individual user.
- The synonyms of a synonym model can only be changed by the owner (user ID) of the model.
- If a user (but not the owner) wants to change the synonyms of the synonym model specified in the provided user profile, z/VSE does not allow it. However, z/VSE creates a copy of the model and allows the user to change the copied synonyms. The user can then use the private synonyms but no longer those of the synonym model. Only if all private synonyms are deleted, the synonyms of the synonym model become accessible again.

Assigning synonym models is a useful method if you have many users that access the same panels and functions. By allowing only the administrator to maintain such models, you can keep control of the synonyms used at your system.

Password Expiration

User IDs should be defined with an expiration date for the password. If a password expires in seven days or less and a user signs-on, the system displays the following message after sign-on:

```
YOUR PASSWORD EXPIRES IN x DAYS
```

In the message, x specifies the number of days before the password expires.

It is recommended that the user changes the password during the next sign on.

In addition, the person who is responsible for maintaining user profiles, usually the system administrator, can change a password using the *Maintain User Profiles* dialog. The dialog is described under "Maintaining User Profiles" on page 98.

How the Password History Is Stored

z/VSE retains the last **twelve** passwords for each user, and prevents these passwords from being reused. If a user then reenters a password that was last used "thirteen passwords previously", z/VSE will permit this previously-used password.

If the user changes his/her password during the sign-on procedure, z/VSE will check the password history. z/VSE does *not* check the password history if the user changes his/her password using the:

- Batch facility IESUPDCF.
- *User Profile Maintenance* dialog.

Please note that you cannot change the rules for the number of passwords that are stored in the password history.

Changing the Minimum Password Length and Revoke Details

From VSE/ESA 2.7 onwards, you can use the IESIRCVT program of the Basic Security Manager (BSM) to define or change:

- The minimum length that a user must enter for a password (between 3 and 8 characters). This applies to password changes that are made during the signon process. However, the z/VSE administrator may still change passwords to a length which is less than the minimum, by using the USER definition tool. This is available in dialog or batch.
- The password warning interval. This is the number of days (between 0 and 9) before a password expires, that the user will be warned that his/her password will expire. This warning is issued immediately after each signon.
- The number of invalid sign-on attempts that a user can make when entering his/her password during sign-on. When this number has been exceeded, the password will be revoked.

The IESIRCVT program is usually executed during system startup, in the procedure USERBG. The signon program IESIES01 will use the details provided by the security manager. Here is an example of how IESIRCVT is used:

```
// EXEC IESIRCVT
 PASSWORD(LENGTH(5))
 PASSWORD(WARNING(3))
 PASSWORD(REVOKE(4))
/*
```

If you specify a REVOKE number-of-days using IESIRCVT, z/VSE will use this value in preference to any value that might have been specified using IESELOGO (which is described in “Setting a Limit for Invalid Sign-On Attempts” on page 332).

Resetting a Revoked User ID

The z/VSE Basic Security Manager (BSM) revokes a user ID if the number of invalid sign-on attempts exceeds a specified limit. This limit for sign-on attempts is specified in the skeleton IESELOGO. Refer to “Setting a Limit for Invalid Sign-On Attempts” on page 332 for detailed information.

Once a user ID is revoked, only a user with system administrator authority can reset it. To reset a revoked user ID, use the dialog *Maintain User Profiles*. See Figure 33 on page 100 for details. In the field REVOKE DATE specify the appropriate date, or set it to zero if the user should never be revoked.

To reset a revoked user ID, you can also use the batch utility program IESUPDCF. You achieve this with the ALTER command and Revoke 0 for the user to be reset. An example now given:

```
* $$ JOB JNM=UPDATECF,CLASS=0,DISP=D
* $$ PUN DISP=I
// JOB UPDATECF
// PAUSE DISCONNECT DTSFILE, IESCNTL
// EXEC PROC=DTRICCF
// DLBL IESCNTL,'VSE.CONTROL.FILE',,VSAM,CAT=VSESPUC
// EXEC IESUPDCF,SIZE=64K
ICCF=YES
             ALT SYSA,PWD=TEEM01,REVOKE=0
/*
/&
* $$ EOJ
```

Chapter 7. Protecting Resources

This chapter describes the z/VSE security support for protecting resources.

It consists of these main sections:

Before reading this chapter you should first read the chapter "Security Support" in the *z/VSE Planning* manual, which provides an overview of the available z/VSE security support.

General Aspects

If you are an experienced z/VSE user, you may skip the following text and turn directly to "z/VSE Security Support" on page 131.

The information stored in a data processing system is often of vital importance to the organization which uses the system. On one hand, the information is necessary for the members of the organization to do their work. On the other hand, the system may store confidential information, whose disclosure to unauthorized persons could mean considerable damage to an organization.

Corresponding to these two categories of information are two aspects of data protection:

Data Security which means protecting information from unauthorized access and use.

Data Integrity which means protecting information from loss or destruction.

The z/VSE Access Control Function, which is the primary topic of this chapter, solely addresses the first aspect: data security. The Access Control Function is part of the Basic Security Manager of z/VSE.

A computer system operating under control of z/VSE offers a number of protection functions for hardware and software. This chapter deals with the

software protection functions provided by z/VSE. For hardware protection functions, refer to the appropriate manuals of the hardware you are using.

Security Considerations

A z/VSE system is designed to protect users' data or applications from interference by other users or applications. However, people who deliberately use their knowledge of the internals of the system can gain unauthorized access to data and resources of the system despite built-in security safeguards. Management is responsible for introducing administrative and operational safeguards that help to avoid such exploitations and that ensure system security to a great extent.

To achieve an acceptable level of system security for a z/VSE installation a user should:

- Ensure that knowledgeable and skilled members of the installation's staff will have little or no chance to access certain data or to use or manipulate certain programs;
- Ensure that resources (mainly programs) that can be used to bypass existing security safeguards are protected properly.
- Ensure that IBM's Diagnosis Reference manuals are given only to a limited set of persons.

The Security Administrator

A security administrator must have adequate software protection functions available to meet all security demands. A person appointed to assume the responsibility of safeguarding installation's assets (data and programs) should, therefore, carefully review the available standard functions. A security administrator should do this review in full consideration of the various responsibilities that a person in such a position normally must assume, some of which are indicated below:

- Together with management, prepare a list of sensitive files and of programs that process data in these files.
- Determine who of the installation's staff is authorized to use those programs and data and establish procedures that ensure that authorized persons only, and no one else, will be able to invoke the programs.
- Protect system libraries adequately.
- Minimize the chance that unauthorized access of sensitive data remains undetected.
- Keeping track of the usage of certain protected programs and data (*logging and reporting*).
- Utilize (except for z/VSE Access Control) special security functions of z/VSE components and programs.

Passwords and User IDs

One of the responsibilities of the security administrator is to assign passwords to users and force the users to change them from time to time in order to avoid damage as a result of inadvertent or intentional disclosure.

The password itself should be composed of a random combination of alphanumeric characters. It should not contain any information or be mnemonic.

If passwords must be included in the job stream, special protective measures may be required. For example, the member containing the job stream could be protected such that unauthorized persons are unable to read it.

General Concept of Access Control

The z/VSE security support allows you to introduce access control at your installation and to implement an acceptable degree of data security. It helps you meet requirements of personal accountability and provides support for:

- **User Identification and Authentication** For details refer to "User Identification and Authentication" on page 145.
- **Access Authorization** For details refer to "DTSECTAB Resources and Access Rights" on page 151, and "Protecting CICS Transactions with Access Control Table DTSECTXN" on page 135.
- **Logging and Reporting**

  For *Logging and Reporting*, the z/VSE optional program VSE/Access Control - Logging and Reporting is available. Its functions are briefly described in "Logging and Reporting Accesses to DTSECTAB Resources" on page 178.

z/VSE Security Support

z/VSE's security support was introduced when CICS/VSE was replaced by the **CICS Transaction Server**. This resulted in changes to the security concept of z/VSE mainly because the CICS Transaction Server does not provide all the security functions that were available with CICS/VSE.

z/VSE provides Secure Sockets Layer (SSL) support for Internet servers and clients.

Note: This support is independent of the security support described in this chapter (except that the related VSE crypto library CRYPTO.KEYRING is defined in DTSECTAB for protection).

The SSL support is documented in detail in the following manuals:
- *z/VSE e-business Connectors User's Guide*
- *CICS Transaction Server for VSE/ESA, Enhancements Guide*

A security repository VSE.BSTCNTL.FILE is also allocated and prepared by z/VSE. This repository is for IBM future use only.

Overview

z/VSE offers two ways of implementing security at a customer's installation:

- Through the Basic Security Manager (BSM) which is part of z/VSE.
- Through an External Security Manager (ESM) which is usually a priced vendor product and which must be installed separately.

This manual discusses mainly the **BSM** and the **basic security support** it provides. Consult also the *z/VSE Planning* manual for further concept and planning information on BSM and ESM.

The major characteristics of the BSM are:

- It provides security support for sign-on, for CICS transactions defined in DTSECTXN, and for resources defined in DTSECTAB.
- To provide this support, the BSM requires a Security Server for security checking. The Security Server runs per default in the FB partition and is always active.

Sign-on Security

This security check involves user ID and password. Examples are:

- Signing-on to z/VSE via the Interactive Interface by entering user ID and password.
- Submitting a job for processing by providing user ID and password through the VSE/POWER JOB statement or the JCL ID statement.

Refer to "User Identification and Authentication" on page 145 for details.

CICS Transaction Security

CICS transaction security is provided for the CICS Transaction Server (not CICS/VSE) via an access control table: DTSECTXN. It defines CICS transactions and their security class. The security class authorizes particular users to access a transaction.

Refer to "Protecting CICS Transactions with Access Control Table DTSECTXN" on page 135 for details.

Access Control for DTSECTAB Resources

The following resources can be defined in DTSECTAB:

- Files
- VSE Libraries
- VSE Sublibraries
- VSE Members

Notes:

1. The access control concept (introduced with VSE/ESA 2.4) has not changed for DTSECTAB and the resources defined in it.
2. DTSECTAB does not include user entries except for the users FORSEC and DUMMY.

Refer to "Protecting Resources with Access Control Table DTSECTAB" on page 144 for details.

Implementation Details

The following list provides technical details of how the security support is implemented:

- The IPL SYS command contains the following security parameters.

  ```
  SEC=NO
  SEC=YES
  SEC=(YES,NOTAPE)
  ESM=name
  SERVPART=partition
  SEC=RECOVER
  ```

  These changes are reflected in the *Tailor IPL Procedure* dialog. Refer to "Using the Tailor IPL Procedure Dialog to Modify Security Parameters" on page 133 for details.
- Security-related **user profile** information is no longer stored in DTSECTAB but in the VSE.CONTROL.FILE.
- The security table DTSECTAB includes resource definitions only, except for the predefined users DUMMY and FORSEC.
- Accesses to the VSE.CONTROL.FILE are through the Security Server. The partition in which the Security Server is to run can be changed with the

SERVPART parameter of the IPL SYS command. Refer also to "Using the Tailor IPL Procedure Dialog to Modify Security Parameters."

- The *Maintain User Profiles* dialog has been enhanced for defining additional security-related information in user profiles such as access control classes and access control rights.
- The parameter AUTH for identifying a user as security administrator is no longer available. Instead, when defining a user profile for a type 1 user (system administrator), this user has automatically ″AUTH authorization″ and can access all resources with access right ALT (Alter).

Implementing the z/VSE Security Support

Basic Security Manager:

Note that the Basic Security Manager (BSM) is always activated during startup, independent of the SEC setting in the IPL SYS command, in order to provide:

- Sign-on security (signing on via the Interactive Interface), and
- CICS transaction security (DTSECTXN).

During initial installation you are asked whether you want to run your system with ″security on″. If you respond with YES, this will set SEC=NO to SEC=YES in the IPL SYS command and provides in addition

- Access control for resources defined in DTSECTAB.

The Security Server required by the BSM runs per default in the FB partition. You can select another partition through the SERVPART parameter of the IPL SYS command.

External Security Manager:

If you want to use an **ESM**, you must define the name of the ESM initialization routine in the ESM parameter of the IPL SYS command. z/VSE always checks for the ESM parameter setting first. If the parameter is specified, the ESM is activated, otherwise the BSM.

If an ESM requires also a security server partition, you can use the SERVPART parameter of the IPL SYS command to define a partition. The default is the FB partition.

Using the Tailor IPL Procedure Dialog to Modify Security Parameters

With the *Tailor IPL Procedure* dialog you can modify the security parameters `SEC`, `ESM`, and `SERVPART` of the IPL SYS command.

To access the dialog, start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
4 (Hardware Configuration and IPL)
3 (Tailor IPL Procedure)

Select the IPL procedure you want to modify and press enter.

```
 TAS$ICM1             TAILOR IPL PROCEDURE: SYS COMMAND

 Enter the required data and press PF5=PROCESS

BUFLD........................ 1           Load printer buffers? 1=yes, 2=no
CHANQ........................ _____       Number of channel queue entries
DASDFP....................... 2           DASD file protection? 1=yes, 2=no
JA........................... 1           Job accounting? 1=yes, 2=no
SUBLIB....................... ____        Number of sublibraries

VMCF......................... _           CMS-VSE console interface? 1=yes,
                                          2=no, or blank for system default

SEC.......................... 2           Access control security? 1=YES, 2=NO,
                                          3=NOTAPE, 4=RECOVER
ESM.......................... ________    Name of the ESM initialization phase
SERVPART..................... FB          Security server partition (F1,F2,...FB)



 PF1=HELP       2=REDISPLAY  3=END                    5=PROCESS
                8=FORWARD
```

Figure 47. Tailor IPL Procedure Dialog

SEC
Specifies whetherDTSECTAB security is to be activated or not.

The following selections are possible. If `1 = YES` is specified, the system performs access authorization checking for resources defined in DTSECTAB. If `2 = NO` is specified, access control for resources defined in DTSECTAB is **inactive** (job control ID card, for example, is ignored). CICS transaction and sign-on security, however, is still **active**. If `3 = NOTAPE` is specified, access control is restricted to DASD files and libraries defined in DTSECTAB. `4 = RECOVER` prevents activation of a security manager. It should be used for recovery actions only, which cannot be done while a security manager is active.

ESM
Specifies the name of an ESM (External Security Manager) initialization phase. If nothing is specified, the BSM (Basic Security Manager) is activated.

SERVPART
Specifies the static partition to be used for the Security Server; the default is FB. Be careful when selecting another partition for the Security Server, which is not recommended. It must be a static partition which is not controlled by VSE/POWER and needs a corresponding priority.

Protecting CICS Transactions with Access Control Table DTSECTXN

CICS transaction security is part of the basic security support (BSM) and protects CICS transactions from unauthorized access if they are specified in table DTSECTXN. This check is always active independent of the SEC setting in the IPL SYS command.

CICS does not allow access to transactions which are not defined in DTSECTXN.

The BSM compares the **security class** of a transaction with the transaction **security key(s)** defined for a particular user in the corresponding user profile. Refer also to the panel ″Add or Change Resource Access Rights″ that is shown in Figure 36 on page 105.

Attempts to invoke CICS transactions by unauthorized users are logged on the system console.

You can define entries in DTSECTXN through either the dialog or macro provided.

Using the Define Transaction Security Dialog

You can add, delete or alter security entries for transactions with the Transaction Security Dialog. These entries are contained in table DTRISEC.Z which is located in IJSYSRS.SYSLIB.

To access the dialog, start with the *z/VSE Function Selection* panel and select:
2 (Resource Definition)
8 (Define Transaction Security)

The *Define Transaction Security* dialog appears where you can press ENTER to list all available security entries, or specify the first characters of the CICS transaction names or the CICS region you want to be listed.

```
 TAS$SECF         DEFINE TRANSACTION SECURITY: SPECIFY FILTER
 Enter the required data and press ENTER.


Press ENTER to list all security entries.

Specify the prefix of the CICS transaction names or the CICS region you
want to be listed and press the ENTER key.


TRANSID............. ____        Enter the full transaction name or
                                 1 - 3 prefix characters, e.g. AB for
                                 all transactions starting with AB

CICS REGION.......... ________   Enter the CICS region.




 PF1=HELP       2=REDISPLAY  3=END                       6=MERGE
```

Figure 48. Define Transaction Security Dialog

TRANSID
Specifies the transaction name, or 1 to 3 prefix characters for a group of CICS transactions that are to be protected and which have the same prefix.

CICS REGION
Specifies the CICS region for which the security entries are to be listed. The name of the CICS region is the user ID under which CICS has been started; for example, DBDCCICS or PRODCICS. This user ID is provided in the ID statement of the CICS startup job, independent of the settings of SYS SEC= in the IPL SYS command. The user ID must be defined in the VSE Control File. In a predefined z/VSE system this user ID is the same as the APPLID of this CICS system which is defined in the SIT (System Initialization Table). The list of entries will be sorted by transaction name in alphabetical order.

Note: To activate the checking of the CICS region, you must specify SECPRFX=YES in the SIT.

Using **PF6=MERGE**, you can merge the new or changed system entries with the security table DTRISEC.Z in IJSYSRS.SYSLIB. This may be useful if you created your own member, or if you applied service, or if you did a Fast Service Upgrade (FSU) to z/VSE 3.1, for example.

After pressing ENTER, the panel *Define Transaction Security* appears. It is a FULIST that displays selected security entries where you can add, alter, or delete entries.

You can include your own lower-case transaction security definitions in the IBM-supplied default member DTSECTXM.A (displayed in Figure 49). These definitions are included in the assembly before the security definitions of the dialog, but they are not shown in the dialog.

```
TAS$SEC1                 DEFINE TRANSACTION SECURITY

Enter the required data and press ENTER.

OPTIONS: 1 = ADD   2 = ALTER    5 = DELETE

 OPT       TRANSACTION NAME  CICS REGION   SECURITY CLASS  GENERIC
  _           AADD                           1
  _           ABRW                           1
  _           ACCT                           1
  _           ACEL                           1
  _           ACLG                           1
  _           AC01                           1
  _           AC02                           1
  _           AC03                           1
  _           AC05                           1
  _           AC06                           1

 LOCATE TRANSACTION NAME == >  ____
 INCLUDE MEMBER == >           IJSYSRS.SYSLIB.DTSECTXM.A
PF1=HELP       2=REDISPLAY  3=END                    5=PROCESS
               8=FORWARD
```

Figure 49. Define Transaction Security Dialog

Fill in the required data and press ENTER to check the values. Press **PF5=PROCESS** to process the changes and to submit the batch job to create table DTSECTXN including the entries defined. The source code is provided in DTSECTXS.A. Do not change this source code.

Generic Transaction Names

An **X** in column GENERIC indicates that this transaction name is interpreted as a generic name. In the example, there exists an explicit definition of security class 30 for transaction AB1, and a generic definition of security class 10 for AB. This means that security class 30 is valid for transaction AB1, and security class 10 for all those transactions that start with AB and do not have an explicit security definition like AB1.

Explanation of INCLUDE MEMBER Field

The job that is created contains an SLI statement for the member name. The member name is specified using its fully-qualified member name. This member contains lower-case transaction security definitions. These definitions are included in the assembly job in front of the security definitions of the dialog. However, these lower-case transaction security definitions are not displayed by the dialog.

The default member is IJSYSRS.SYSLIB.DTSECTXM.A.

Merging, Processing and Activating DTSECTXN

The MERGE function of the dialog can merge member DTRISEC.Z in IJSYSRS.SYSLIB with one or more of the following members:

- DTRISEC.U

  This member is created by the system in case of FSU or the application of service.
- A user-created member, for example, USERMEMB.Z in userlib.usersublib.

  The member must be of type Z and must have the special table format required by the dialog.

Example

You want to create a new DTSECTXN by merging macro member MYMAC.A in userlib.usersublib with DTRISEC.Z. The following steps are required:

1. Run a job stream for MYMAC.A that includes the following command to create the required table format:
   ```
   EXEC REXX=IPFTABLE,PARM='ULIB.USUBLIB.MYMAC.A ULIB.USUBLIB.MYTAB.Z'
   ```
2. Select the MERGE function of the dialog and enter your parameters (member name and sublibrary name) to merge your table with DTRISEC.Z in IJSYSRES.SYSLIB.
3. Select the PROCESS function of the dialog to catalog the new phase DTSECTXN into IJSYSRS.SYSLIB and PRD2.SAVE. In addition, macro DTSECTXS.A is cataloged into PRD2.CONFIG. Both, DTSECTXN and DTSECTS.A include the changes of MYTAB.Z.

   The PROCESS function:
   - Creates job CATSEC which you must submit for processing.
   - Updates table DTRISEC.Z which is used when the dialog is called the next time.
4. DTSECTXN is activated with the following command:
   ```
   CEMT PERFORM SECURITY REBUILD
   ```
   The command is issued by the job generated. You may issue this command directly if the update did not complete correctly. DTSECTXN is also activated by stopping/starting the CICS partition.

Using the Macro DTSECTXN

The DTSECTXN macro can be used to build the CICS transaction security table DTSECTXN. For each transaction to be defined, a security class must be assigned.

Syntax Diagram:

```
►►──DTSECTXN NAME=──┬────────────────────┬──transid,TRANSEC=(class)──────────►
                    └─cics_region_prefix.┘

►──┬────────────────┬──┬────────────────────────┬──────────────────────────────►◄
   └─,TYPE=─GENERIC─┘  └─,SUBTYPE=──┬─INITIAL─┬─┘
                                    └─FINAL───┘
```

Refer also to the definition of CICS REGION on the preceding page.

Parameter Description:

NAME=cics_region_prefix.transid
The cics_region_prefix is a 1 to 8-character identifier, and transid specifies a 1 to 4-character transaction name. Generic transaction names are allowed if TYPE=GENERIC has been specified. A transaction name may include alphabetic and numeric characters, and the following special characters:

`$ # @ / % ¢ ? ! : | " = ¬ ; < >`

Mixed case transaction names are supported.

The cics_region_prefix can be used to identify a particular CICS subsystem if two or more CICS subsystems run under z/VSE.

TRANSEC=(class)
Defines the security class (1 to 64) to control the invocation of a transaction. For example, if TRANSEC=(5) is specified, only users defined with security key 5 in their profile can start this transaction.

Only one security class can be specified for a transaction.

Security keys for a user profile can only be specified via the *Maintain User Profiles* dialog, which is described in Chapter "Maintaining User Profiles" on page 98.

TYPE=GENERIC
This parameter is optional and causes a transaction name (transid) to be interpreted as a generic name. This means that an entry with TYPE=GENERIC covers all transactions whose names start with the characters defined as *transid*, except such a name is defined explicitly (refer also to "Generic Transaction Names" on page 136). Any abbreviation is accepted: TYPE=G, TYPE=GE, and so on.

SUBTYPE=INITIAL or FINAL
The parameter is required in the first and the last call of the DTSECTXN macro, and is not allowed in the calls between. The first macro call must include SUBTYPE=INITIAL, and the last one SUBTYPE=FINAL.

Examples:

```
DTSECTXN NAME=DBDCCICS.AADD,TRANSEC=(1)
DTSECTXN NAME=TR,TRANSEC=(1),TYPE=GENERIC
```

Example of the CICS Transaction Security Table DTSECTXM

As shipped, z/VSE provides a predefined DTSECTXM. DTSECTXM includes entries such as the following (it does not include generic entries):

```
DTSECTXN NAME=emai,TRANSEC=(1),SUBTYPE=INITIAL
DTSECTXN NAME=ftp,TRANSEC=(1)
DTSECTXN NAME=iccf,TRANSEC=(1)
DTSECTXN NAME=lpr,TRANSEC=(1)
DTSECTXN NAME=newc,TRANSEC=(1)
DTSECTXN NAME=ping,TRANSEC=(1)
DTSECTXN NAME=ropc,TRANSEC=(1)
DTSECTXN NAME=teln,TRANSEC=(1)
```

Figure 50. Extract of Table DTSECTXM

The above table contains transaction security definitions written in *lower case*. You can enter your own transaction security definitions in this table. These definitions are included in the assembly job in front of the security definitions of the dialog. However, these lower-case transaction security definitions are not displayed by the dialog.

Example of the CICS Transaction Security Table DTSECTXN

As shipped, z/VSE provides a predefined DTSECTXN. DTSECTXN includes entries such as the following (it does not include generic entries).

```
DTSECTXN NAME=AADD,TRANSEC=(1)
DTSECTXN NAME=ABRW,TRANSEC=(1)
DTSECTXN NAME=ACCT,TRANSEC=(1)
DTSECTXN NAME=ACEL,TRANSEC=(1)
DTSECTXN NAME=ACLG,TRANSEC=(1)
DTSECTXN NAME=AC01,TRANSEC=(1)
DTSECTXN NAME=AC02,TRANSEC=(1)
DTSECTXN NAME=AC03,TRANSEC=(1)
DTSECTXN NAME=AC05,TRANSEC=(1)
DTSECTXN NAME=AC06,TRANSEC=(1)
DTSECTXN NAME=AC2A,TRANSEC=(1)
DTSECTXN NAME=AC2C,TRANSEC=(1)
DTSECTXN NAME=AC2D,TRANSEC=(1)
DTSECTXN NAME=AC2E,TRANSEC=(1)
DTSECTXN NAME=AC2F,TRANSEC=(1)
DTSECTXN NAME=AC20,TRANSEC=(1)
DTSECTXN NAME=AC21,TRANSEC=(1)
DTSECTXN NAME=AC22,TRANSEC=(1)
DTSECTXN NAME=AC23,TRANSEC=(1)
DTSECTXN NAME=AC24,TRANSEC=(1)
DTSECTXN NAME=AC25,TRANSEC=(1)
```

⋮

```
DTSECTXN NAME=XPLA,
DTSECTXN NAME=2RPS,
DTSECTXN NAME=8888,
DTSECTXN NAME=9999,TRANSEC=(1),SUBTYPE=FINAL
END
```

Figure 51. Extract of Table DTSECTXN (Source Format DTSECTXS)

Migrating TRANSEC Values from CICS/VSE to the CICS Transaction Server

The CICS Transaction Server uses security table DTSECTXN to protect transactions against unauthorized access. In CICS/VSE, a transaction is protected with the TRANSEC parameter in the Program Control Table(s) (PCT). z/VSE provides migration support to convert **TRANSEC** values from CICS/VSE into entries in transaction security table **DTSECTXN** of the CICS Transaction Server.

This support is available for the Basic Security Manager (BSM) only.

For migration support, z/VSE provides three REXX/VSE procedures (DTSECTXS, DTSECTX2, and DTSECTX3) which are stored as skeletons in VSE/ICCF library 59, named **SKSECTXS**, **SKSECTX2** and **SKSECTX3**. The three procedures allow three different ways of migrating CICS/VSE TRANSEC values.

1. Procedure **DTSECTXS** (skeleton SKSECTXS)

   Procedure DTSECTXS uses as input existing library members that include Program Control Table (PCT) definitions and generates output for transaction security table DTSECTXN of the CICS Transaction Server. Examples of library members with PCT definitions are the CICS/VSE tables IESZPCT and DFHPCTxx.
2. Procedure **DTSECTX2** (skeleton SKSECTX2)

   Procedure DTSECTX2 uses as input the VSE/VSAM output file DFHXSMA created by the Security Migration Aid (SMA) of CICS/VSE. The migration aid extracts security information from CICS/VSE and the VSE.CONTROL.FILE and stores it in file DFHXSMA. DTSECTX2 reads the PCT information from DFHXSMA and generates output for transaction security table DTSECTXN of the CICS Transaction Server.
3. Procedure **DTSECTX3** (skeleton SKSECTX3)

   Procedure DTSECTX3 uses as input the CICS/VSE DFHCSDUP definitions of format DEFINE TRANSACTION(xxxx) ... TRANSEC(yy) stored in a library member.

The procedures create as output a library member whose name is to be defined in the OUTFILE parameter. Depending on the FORMAT parameter, the procedures create one of the following:

- A table in IPF format.

  This format should be selected if the MERGE function of the *Define Transaction Security* dialog is to be used. With this function, existing transaction security definitions in DTSECTXN can be merged with the migrated security definitions from CICS/VSE. Note that in this case a migrated security definition for a particular transaction overwrites an already existing definition for that transaction in DTSECTXN.
- A file consisting of DTSECTXN macro definitions.

  This file can be used as input for a DTSECTXN assembly run.

To use skeleton SKSECTXS, SKSECTX2, or SKSECTX3 proceed as follows:

1. Get a copy of the skeleton from VSE/ICCF library 59.
2. Modify the parameters as required. Parameters can be specified in the PARM operand, via SYSIPT, or in a combination of both. Refer to “Using Procedure DTSECTXS for Migration” on page 141, to “Using Procedure DTSECTX2 for Migration” on page 142, or to “Using Procedure DTSECTX3 for Migration” on page 143 for a description of the parameters and examples of how to use them.

3. Submit the job (skeleton) for processing. The job catalogs first procedure DTSECTXS, DTSECTX2, or DTSECTX3 into PRD2.CONFIG and invokes it from there. You may change the name of the sublibrary used for cataloging.
4. Verify your output file and the printout in the LST queue (job DTSECTXS, DTSECTX2, or DTSECTX3) to ensure correct processing.

Using Procedure DTSECTXS for Migration

DTSECTXS Parameters

DTSECTXS uses the following parameters:

Parameter **Description**

INFILE Defines the library member name of the file containing PCT definitions with TRANSEC values.

FORMAT Defines the format of the output file; either IPF or MACRO.

- **IPF** is the format required when using the MERGE function of the *Define Transaction Security* dialog to merge existing security entries in DTSECTXN with migrated entries. This is the default.
- **MACRO** is the format consisting of DTSECTXN macro definitions.

OUTFILE Defines the library member name of the file to be created containing the DTSECTXN definitions. The default is PRD2.CONFIG.DTRISEC.M for the IPF format and PRD2.CONFIG.DTSECTXS.A for the MACRO format.

REGION Defines the name of the CICS-Region to be included in the DTSECTXN definitions. If region names should not be included in the DTSECTXN definitions, omit this parameter or specify REGION= followed by a blank.

Invoking DTSECTXS

The following example invokes DTSECTXS and uses the PARM operand to provide parameters:

```
// EXEC REXX=DTSECTXS,PARM='INFILE=IJSYSRS.SYSLIB.IESZPCT.A'
```

For the FORMAT, OUTFILE, and REGION parameters the defaults are effective.

The following example invokes DTSECTXS and uses SYSIPT to provide parameters:

```
// EXEC REXX=DTSECTXS
INFILE=IJSYSRS.SYSLIB.IESZPCT.A
FORMAT=MACRO
OUTFILE=LIB.SLIB.DTSECTXS.A
REGION=DBDCCICS
```

DTSECTXS Return Codes:

DTSECTXS may issue the following return codes:

0	Processing successful
2	Unsupported character in transaction name: <*tr-name*>
4	Syntax error: Invalid keyword <*keyword*>
8	Member <*mem.type*> already exists in sublibrary <*lib.slib*>
12	Reading member <*mem.type*> failed with RC=*n*

16 Transaction name longer than 4 characters: <*tr_name*>
20 TRANSEC definition is not a number: <*tran_sec*>
24 No transaction specifications found
28 Accessing sublibrary <*lib.slib*> failed
32 Writing member <*lib.slib.mem.type*> failed with RC=*n*

Using Procedure DTSECTX2 for Migration

DTSECTX2 Parameters

DTSECTX2 uses the following parameters:

Parameter **Description**

INFILE Defines the VSE/VSAM file created by the CICS/VSE Security Migration Aid (SMA).

FORMAT Defines the format of the output file; either IPF or MACRO.

- **IPF** is the format required when using the MERGE function of the *Define Transaction Security* dialog to merge existing security entries in DTSECTXN with migrated entries. This is the default.
- **MACRO** is the format consisting of DTSECTXN macro definitions.

OUTFILE Defines the library member name of the file to be created containing the DTSECTXN definitions. The default is PRD2.CONFIG.DTRISEC.M for the IPF format and PRD2.CONFIG.DTSECTXS.A for the MACRO format.

CATALOG Defines the name of the VSE/VSAM catalog which includes the input file (INFILE). The default is the master catalog IJSYSCT.

REGION Defines the name of the CICS-Region to be included in the DTSECTXN definitions. Default is the region name found in the input file (INFILE). If region names should not be included in the DTSECTXN definitions, specify REGION= followed by a blank.

Invoking DTSECTX2

The following example invokes DTSECTX2 and uses the PARM operand to provide parameters:

```
// EXEC REXX=DTSECTX2,PARM='INFILE=CICS.DFHXSMA            X
OUTFILE=LIB.SLIB.DTSECTX2.A REGION=DBDCCICS'
```

The following example invokes DTSECTX2 and uses SYSIPT to provide parameters:

```
// EXEC REXX=DTSECTX2
INFILE=CICS.DFHXSMA
CATALOG=IJSYSCT
FORMAT=MACRO
OUTFILE=LIB.SLIB.DTSECTX2.A
REGION=DBDCCICS
```

DTSECTX2 Return Codes

DTSECTX2 may issue the following return codes:

0 Processing successful
2 Unsupported character in transaction name: <*tr-name*>
4 Syntax error: Invalid keyword <*keyword*>
8 Member <*mem.type*> already exists in sublibrary <*lib.slib*>
12 IDCAMS PRINT of VSE/VSAM file failed with RC=*n*

16 Transaction name longer than 4 characters: <*tr_name*>
24 No transaction specifications found
28 Accessing sublibrary <*lib.slib*> failed
32 Writing member <*lib.slib.mem.type*> failed with RC=*n*

Using Procedure DTSECTX3 for Migration

DTSECTX3 Parameters

DTSECTX3 uses the following parameters:

Parameter **Description**

INFILE Defines the name of the file (name used by Librarian) containing the DFHCSDUP definitions.

FORMAT Defines the format of the output file; either IPF or MACRO.

- **IPF** is the format required when using the MERGE function of the *Define Transaction Security* dialog to merge existing security entries in DTSECTXN with migrated entries. This is the default.
- **MACRO** is the format consisting of DTSECTXN macro definitions.

OUTFILE Defines the library member name of the file to be created containing the DTSECTXN definitions. The default is PRD2.CONFIG.DTRISEC.M for the IPF format and PRD2.CONFIG.DTSECTXS.A for the MACRO format.

REGION Defines the name of the CICS-Region to be included in the DTSECTXN definitions. Default is the region name found in the input file (INFILE). If region names should not be included in the DTSECTXN definitions, omit this parameter or specify REGION= followed by a blank.

Invoking DTSECTX3

The following example invokes DTSECTX3 and uses the PARM operand to provide parameters:

```
// EXEC REXX=DTSECTX3,PARM='INFILE=LIB.SLIB.MYTRANS.DEFS             X
OUTFILE=LIB.SLIB.DTSECTX3.A REGION=DBDCCICS'
```

The following example invokes DTSECTX3 and uses SYSIPT to provide parameters:

```
// EXEC REXX=DTSECTX3
INFILE=LIB.SLIB.MYTRANS.DEFS
CATALOG=IJSYSCT
FORMAT=MACRO
OUTFILE=LIB.SLIB.DTSECTX3.A
REGION=DBDCCICS
```

DTSECTX3 Return Codes

DTSECTX3 may issue the following return codes:
0 Processing successful
2 Unsupported character in transaction name: <*tr-name*>
4 Syntax error: Invalid keyword <*keyword*>
8 Member <*mem.type*> already exists in sublibrary <*lib.slib*>
12 Reading member <*mem.type*> failed with RC=*n*
16 Transaction name longer than 4 characters: <*tr_name*>
20 TRANSEC definition is not a number: <*transec*>

24 No transaction specifications found
28 Accessing sublibrary <*lib.slib*> failed
32 Writing member <*lib.slib.mem.type*> failed with RC=*n*

Protecting Resources with Access Control Table DTSECTAB

The z/VSE Access Control Function, which is part of the Basic Security Manager (BSM), controls the access to the following resources:

Files

Libraries

Sublibraries

Members

These resources and their access control parameters must be defined in the access control table **DTSECTAB**. To activate access control for DTSECTAB resources, the security parameter in the IPL SYS command must be set to **SEC=YES**.

Note that the access control concept has not changed for resources (Files, Libraries, Sublibraries, Members) defined in DTSECTAB. However, user profiles are no longer stored in DTSECTAB but in the VSE.CONTROL.FILE.

Security Checking

Security checking is done on two distinct levels:

1. User **identification** and **authentication**:
   - User identification: this is done by checking the **user ID**. Is this user ID known to the system?
   - User authentication: is the user really the person that owns this user ID? This is checked either via an **explicit password** supplied with a job, or it is checked via an indication that the password had been validated at some earlier stage. This may have been done during sign-on, for example, before the job was submitted. In this case, no further password check is necessary.
2. **Access authorization**:

   is the user permitted to **access a particular** resource such as a file, library, sublibrary, or member?

   This is done by comparing the
   - **User profile** information in the VSE.CONTROL.FILE with the
   - **Resource profile** information in DTSECTAB.

User Profiles

User profiles are stored in the VSE.CONTROL.FILE. A user profile specifies, for an individual user, the **access rights** to resources. You define user profiles via the Maintain User Profile Dialog. See "Maintaining User Profiles" on page 98 for details.

The ACC parameter defines access control **classes** together with associated **access rights**. A user's access control **class** can have one of the following access rights:

- ALT (Alter)
- UPD (Update)
- READ (Read-only)
- CON (Connect)

These access rights are ordered hierarchically: ALTER implies UPDATE, UPDATE implies READ, READ implies CONNECT.

The definition of access rights will be discussed in detail in “Access Rights” on page 151. Refer also to Figure 36 on page 105 where the ″Add or Change Resource Access Rights″ panel of the *Maintain User Profiles* dialog is shown.

A user that is defined as system administrator (type 1 user) in the user profile has unrestricted access with access right of ALT to all protected resources.

Resource Definitions

Batch resource definitions are stored in the security table DTSECTAB.

For each resource to be protected, the security administrator defines one or more access control classes in the corresponding resource profile. In general, resources without an entry in DTSECTAB are not protected.

Access control classes are numbers between 1 and 32 which are assigned to the resource.

The following z/VSE resources can be protected:
- Files
- Libraries
- Sublibraries
- Members

A typical definition might look as follows:

```
DTSECTAB TYPE=SUBLIB,NAME=AUX.PR$302,ACC=(8,9)
```

Sublibrary AUX.PR$302 is defined as a resource that can be accessed by users who have access control classes 8 and 9 defined in their user profile.

Authorization of a particular user to access a resource is determined by a match of the access control classes. In our example “An Example” on page 154, user ENDU with access control class 1 through 8 is allowed to access sublibrary AUX.PR$302 due to the match on class 8. The access right is limited to UPDATE. An attempt by ENDU to ALTER (rename or delete) the sublibrary would be an access violation.

Access Control via classes establishes an **individual access right** for the user. For resources ″library, sublibrary, and member″, additionally a **universal access right (UACC)** can be specified. It grants **all** users of the system, irrespective of the classes specified in user profiles (if such profiles exist), the defined access right of ALT, UPD, READ or CON. For a resource with a **universal** access right, **individual** access rights are meaningful only if they are higher than the universal access right, because at least the UACC will be granted to any user.

You find detailed information about protection of resources in “DTSECTAB Resources and Access Rights” on page 151.

User Identification and Authentication

In a secured z/VSE system, batch jobs that are submitted for processing are checked for identification and authentication (security identification):
- User identification: is the user known to the system? That is, does the user have a user profile in the VSE.CONTROL.FILE?

- Authentication: is the user really the person that owns this user ID?

Security identification is supplied in three ways:

1. During sign-on to a z/VSE subsystem such as the z/VSE Interactive interface or VSE/ICCF. Jobs submitted from here run under the sign-on user ID.
2. An explicit security identification in the SEC parameter of the VSE/POWER JECL statement * $$ JOB of a submitted job.

   An equivalent identification can be given for jobs submitted via a z/VSE-internal interface, the "VSE/POWER Spool Access Support". It is described in the IBM manual *VSE/POWER Application Programming* under "Introduction to Spool-Access Support".
3. An explicit security identification in the // ID job control statement of a job submitted.

If you submit a job, you need not explicitly enter user ID and password with each submission, as explained in section "Authenticated Jobs" on page 147.

Security Information in the JECL Statement * $$ JOB

The parameter SEC in the * $$ JOB statement of VSE/POWER specifies user ID and password of the VSE/POWER job to be submitted:

* $$ JOB ... SEC=(userid,password)

The SEC parameter is optional. However, if specified, it must contain both user ID and password.

The security information in the * $$ JOB statement is valid for the entire sequence of z/VSE jobs included in a VSE/POWER job stream.

For a complete description of the * $$ JOB statement refer to the manual *VSE/POWER Administration and Operation* under "* $$ JOB: Marking the Start of a VSE/POWER Job".

Security Information in the JCL Statement // ID

The job control statement // ID carries the same information as the JECL statement * $$ JOB, that is: user ID and password.

The information is valid for one z/VSE job: it covers the job where it is included, but not any other job that might follow.

A // ID statement overrides the VSE/POWER security information for the length of the z/VSE job. After that, VSE/POWER's security information becomes effective again.

// ID statements should be avoided because users with access to jobs in the VSE/POWER reader queue can see both, user ID and password. Specifying the user ID and password in the *** $$ JOB** statement is the better and recommended solution. Retrieving a job from the VSE/POWER reader queue generally does not reveal user ID and password.

However, there are situations where the statement is needed, for example

- In z/VSE startup procedures. Please refer to "Access Control for Startup Procedures" on page 160.
- In PAUSExx jobs. Please refer to "Tasks to be Done after Initial Installation" on page 168.

- In jobs that accomplish the transferring of jobs and files between systems. z/VSE dialogs may create a // ID statement if the remote system is at *backlevel* (z/VSE prior to VSE/ESA 1.3). Please refer to “Transfer of Jobs or Files/Members between Systems” on page 150.

Authenticated Jobs

The user ID, which z/VSE knows (for example from sign-on), is sufficient for user authentication if a batch job is submitted from one of the five sources:

1. z/VSE Interactive Interface
2. VSE/ICCF
3. A workstation via the SEND/RECEIVE command interface
4. A job with explicit user ID and password specification
5. Another authenticated job.

Therefore, a user who submits a job from any of the above sources does not need to care about the user ID or the password for this job.

A job that is submitted on behalf of a user whose user ID and password have been validated earlier is called **authenticated job**.

Note: If a job is to run with another user profile than the one of the submitter, user ID and password must be supplied. In this case, not the // ID statement but the *** $$ JOB** statement should be used for the reasons outlined above.

The subject of user ID propagation is discussed in more detail in the section, “Propagation of Security Identification.”

Introducing DTSECTAB Resource Protection

The predefined table DTSECTAB contains mainly system-defined resources. It does not use any classes, the resources are protected via **universal access rights** only. In this way the pregenerated definitions do not interfere with the user’s installation-specific class definitions.

You can build upon this table if you need to tailor the given support. You may, for example, extend the set of resources to include your own resources. Or, you may want to establish a set of **access control classes** to implement your own rules of differentiation between individual users.

More details on the predefined support are given in “Predefined DTSECTAB Security Support” on page 168.

Propagation of Security Identification

The information provided in this section is related to one or more **VSE/POWER batch environments**.

A user who submits a batch job from the z/VSE Interactive Interface (or from a VSE/ICCF terminal or through the SEND command from a workstation) does not have to pass user ID and password for the job submitted. The system automatically makes sure that the job will run with the user’s profile information. Only if the job is to run with another user profile, the submitter must specify the other user’s security identification.

VSE/POWER Authenticated Jobs

A job is considered authenticated if the user ID and password of the submitter were checked successfully before the job was submitted. This type of job thus is called an **authenticated job**. Only the **user ID** of the submitter is associated with the job.

An authenticated job retains its status even when being transferred via

- A PNET network to another VSE/POWER system
- VSE/POWER shared spooling to another system
- A POFFLOAD tape to another system

under the condition that the originating system and the executing system belong to the **same security zone**. (The concept of *security zone* is described below, under "Propagating Security Identification between VSE/POWER Subsystems" on page 149.)

The authenticated job is submitted internally via the

- VSE/POWER Spool Access Support or Extended Device Support (XPCC interface), or
- VSE/POWER spooled punch output (* $$ PUN DISP=I).

The user ID is propagated from the submitting job into the authenticated job.

Examples:

- A job that is generated as * $$ PUN DISP=I output inherits the security identification from the job which generates the punch output.
- The utility DTRIINIT (which loads programs into the RDR queue via Spool Access Support) propagates the security identification of the utility job to the jobs that are being loaded.

If the submitting job contains a // ID statement, the propagated user ID is taken from that statement. Note that when using a VSE/POWER JECL statement * $$ PUN with DISP=I, the // ID statement must precede the * $$ PUN. The submitting job may override the propagated security identification by punching a * $$ JOB statement which contains user ID and password for the job being created.

Note that a job runs even if the job's user ID does not exist in the VSE.CONTROL.FILE, but can access resources with a (sufficient) universal access right, or which are unprotected.

From a system with security inactive (SEC=NO), propagation of security identification is not possible.

Propagating Security Identification between VSE/POWER Subsystems

The CPU on which a batch job is submitted need not be the same as the CPU which is to execute the job. This is the case when a job is submitted via

- A PNET network to another VSE/POWER system
- VSE/POWER shared spooling to another system
- A POFFLOAD tape to another system

For the propagation of security information to an authenticated job, it is important whether or not the two systems belong to the same *security zone*. When an authenticated job runs in a security zone other than the originating zone, it runs without security authorization.

Security Zone

A security zone consists of a group of systems where a given user ID that occurs on any of these systems identifies the same user.

If all user profiles on the submitting system and the executing system are unique (that is: a user ID identifies the same person on each system), the submitter needs not be concerned about passing security identification to the other system because the (local) user ID does not belong to another person on the other system. The security administrator should define the two systems as belonging to the same **security zone**.

Within one security zone, authenticated jobs keep their status. The security zone is defined to VSE/POWER in the SECNODE parameter of either

- the POWER generation macro:

  ```
  POWER SECNODE=zonename
  ```

or the

- VSE/POWER SET statement:

  ```
  SET SECNODE=zonename
  ```

For more details refer to the IBM manual *VSE/POWER Administration and Operation* under “POWER Generation Macro” and under “SET: Setting VSE/POWER Startup Control Values”.

Equal SECNODE names on multiple systems mean: each user ID describes the same person on those systems where the user ID is defined in the VSE.CONTROL.FILE.

If, on the other hand, the security administrator cannot guarantee that user profiles are unique on the two systems, different SECNODE names must be defined on the two systems. In this case, a submitter must explicitly pass along security identification in the * $$ JOB statement in order to access protected resources with insufficient universal access rights in another security zone. This ensures that the job does not run with the user profile of a different person.

General Rules for VSE/POWER Subsystems

The preceding discussion assumed that the submitting system and the executing system are both IPLed with security active (SEC=YES).

Consider the following combinations:

Submitting System SEC=YES - Executing System SEC=YES:

- If the SECNODE IDs are equal, the submitting system propagates the security identification to the executing system unless it is explicitly overridden by the submitter.
- If the SECNODE IDs are not equal, the submitter should explicitly supply security information in the * $$ JOB statement. The implicitly propagated security identification is ignored on the executing system, and further propagation cannot take place.

Submitting System SEC=YES - Executing System SEC=NO:

The security information transferred by the submitting system is ignored.

Submitting System SEC=NO - Executing System SEC=YES:

No security information is implicitly propagated with the submitted job. The submitter should explicitly supply security information in the * $$ JOB statement.

Submitting System SEC=NO - Executing System SEC=NO:

No security information is propagated with the submitted job.

The preceding text outlined the general rules that govern security authorization between systems. The following sections deal with two special VSE/POWER environments:

- Shared Spooling
- Job/File transfer from one system to another.

Security Checking under VSE/POWER Shared Spooling

In a VSE/POWER shared spooling environment, typically the security zone would be equal for all sharing systems. This allows users to have their jobs run on **any** of these systems. If for some special reason any of the shared systems has its own security zone, VSE/POWER's job scheduling will take this into account. VSE/POWER attempts to execute authenticated jobs on systems with matching SECNODE names.

For detailed information (in particular about the SHARED parameter of the POWER macro) refer to the IBM manual *VSE/POWER Administration and Operation* under "POWER Generation Macro".

Transfer of Jobs or Files/Members between Systems

For a VSE/POWER PNET network, z/VSE provides dialogs for

- Submitting a job to another system in the network
- Transferring/retrieving VSAM files or VSE/ICCF members.

When the local system and/or the remote system run with security active, access control also needs to be considered. The system in which you use the dialog and submit the job is called *local system*. In particular, the propagation of security information is affected by

- Differing security zones
- The possible participation of a *backlevel system* (this can be a VSE system prior to VSE/ESA 1.3, or a VSE/SP system).

Under "Submitting Jobs to Other Systems", the IBM manual *z/VSE Networking Support* provides details and describes how you work with the dialog.

DTSECTAB Resources and Access Rights

This section describes:

- The **resources** that can be protected with security table DTSECTAB
- The **access rights** associated with those resources.

Which Resources Can Be Protected in DTSECTAB?

Resources to be protected must be defined in DTSECTAB. The following resources can be protected:

1. All **libraries, sublibraries**, and all their **members**.

   Members are protected at **member name** level. That is, within one sublibrary, members of different types with the same name are protected under the same resource profile. For example, if a user has an access right to member name PROG1, that right applies to PROG1.A, PROG1.E, PROG1.OBJ, PROG1.PHASE, and PROG1.PROC.
2. **Files** as outlined below:
   - All VSE/VSAM KSDS, RRDS, VRDS and ESDS accessed directly via an ACB macro, and all VSE/VSAM-managed SAM files accessed via either a DTFSD or an ACB macro, or by appropriate file definition statements of the IBM compiler(s) used at your installation. Note that the file's VOLSER and catalog are not checked.

     When VSE/VSAM data is accessed via a path, the path name is used for access checking.

     The file's catalog can only be checked if the cluster is defined with the authorization parameter, and a VSE/VSAM *user security verification routine* (USVR) is coded. The catalog name can be passed to the USVR exit after the entry point name in the authorization parameter. You find detailed information on the USVR exit in the IBM manual *VSE/VSAM Commands* under "User Security-Verification Routine".
   - All non-VSAM disk files and standard-labeled tape files that are defined by a file description macro (DTFxx), or by appropriate file definition statements of the IBM compiler(s) used at your installation.

You cannot protect through DTSECTAB entries:

- Unlabeled tapes
- Tapes with non-standard labels
- Diskette files

Access Rights

The following discussions frequently refer to specific access rights. Table 3 on page 152 is a **summary of access rights** in relation to protected resources. Detailed explanations are given in the following section.

Access rights are to be defined for a user's access control classes through the "Maintain User Profiles" dialog.

Table 3. Access Rights for Libraries, Sublibraries and Members

Access Right	Library	Sublibrary	Member
ALT	Create and delete.	Create, delete and rename.	Create, delete and rename.
UPD	Update contents. Create, delete and rename (ALT) sublibraries in it.	Update contents. Catalog, delete and rename (ALT) members in it.	Update contents. Add, delete and change lines.
READ	Read only for library and all sublibraries in it.	Read only for sublibrary and all members in it.	Read only.
CON	Access to sublibraries in it, if user has access right for these sublibraries individually.	Access to members in it, if user has access right for these members individually.	Not Applicable.

The meaning of the access rights is as follows:
- ALT = Alter
- UPD = Update
- READ = Read
- CON = Connect

Please recall that ALT implies UPD, UPD implies READ, READ implies CON (where applicable).

Note: A user must have at least access right CON to a protected sublibrary in order to access it by a LIBDEF statement.

The above table shows access rights for libraries, sublibraries and members. Access rights for **files** are as follows:
- Both ALT and UPD provide the right to **create, delete, rename** a file, and to **add, delete,** and **change** records.
- READ means 'read-only'.
- CON is not applicable to a file.

For **DASD files**, the Access Control Function determines during OPEN processing the required access right depending on the ACB (access control block) of a VSE/VSAM or DTF file. For example, to open a DTFSD file for INPUT, only an access right of READ must be defined in DTSECTAB. If the same file is opened for OUTPUT, an access right of UPD is required in DTSECTAB.

The following table lists the access rights required for DASD files. Note that for DTFDI files access checking is already done when the ASSGN statement is being processed.

Table 4. Access Rights Required for ACB or DTF Open Processing

DASD File:	Access Right Required
ACB; MACRF defines (..,OUT)	UPD
ACB; MACRF does not define (..,OUT)	READ
DTFDA (every case)	UPD
DTFIS (every case)	UPD
DTFPH (every case)	UPD
DTFDI DEVADDR=SYSIPT|SYSRDR	READ
DTFDI DEVADDR=SYSPCH|SYSLST	UPD
DTFSD TYPEFLE=INPUT	READ
DTFSD TYPEFLE=OUTPUT	UPD
DTFSD UPDATE=YES	UPD
DTFSD TYPEFLE=WORK	UPD
DTFSD TYPEFLE=WORKIN	READ
DTFSD TYPEFLE=WORKUP	UPD
DTFSD TYPEFLE=WORKMOD	UPD

Two Kinds of Access Rights

The security administrator, who is defined as type 1 user (system administrator) in the user profile, has the highest access right to all resources.

The following text refers to a user who is **not** the security administrator. For such users, access to a protected resource is controlled by one of the following access rights:

1. Universal Access Right
2. A match of an access control class.

1. **Universal Access Right**

It grants **all users** of the system a particular access right to a library, sublibrary or member (files cannot have a universal access right). A universal access right is defined in the UACC parameter of LIB, SUBLIB or MEMBER-type calls in the DTSECTAB macro. For example, the macro call

```
        DTSECTAB TYPE=MEMBER,                                     C
             NAME=IJSYSRS.SYSLIB.STDLABUP,                        C
             UACC=UPD
```

authorizes all users of the system to update the contents of member STDLABUP in the system sublibrary IJSYSRS.SYSLIB.

Access to a protected resource is allowed if the universal access right of the resource is sufficient for the requested access. In the example above, UACC=UPD is sufficient when a program attempts to read or change member STDLABUP.

2. Access by **Access Control Class**

In a resource profile, one or more of 32 access control classes can be assigned to a resource (ACC parameter of the DTSECTAB macro).

The user profiles also refer to these classes and are defined with the "Maintain User Profile" dialog. In addition, the user profile specifies which access right the user has for a particular class: CON, READ, UPD, or ALT. The ACC parameter thus defines the range of the user's authorization for the specified access class or group of classes as long as the universal access right for the resource is not sufficient.

A resource that has neither a UACC nor an access control class defined can only be accessed by the system administrator (type 1 user).

An Example

The example assumes a type 3 user with the following definitions in the corresponding user profile:

```
USERID=ENDU
PASSWORD=XB3L25
ACCESS CLASS=1-8
ACCESS RIGHT=UPD
```

The access request concerns the following resource:

```
DTSECTAB TYPE=SUBLIB,NAME=AUX.PR$302,ACC=(8,9),UACC=READ
```

The Access Control function first checks whether the resource has a sufficient universal access right. If this is not the case, it compares the user profile entry with the resource profile entry. This check is done in two steps:

1. A check for a match of the **access control class**.

   If there is a match between the access control classes of the user profile entry and the profile entry of the resource to be accessed, processing is allowed to continue. Otherwise, a security violation is indicated. This check is done for the requested resources.
2. A check for the user's **access right** in the user profile and the **type of access** attempted by a job or program.

   If the access right for this class in the user profile is sufficient for the type of access attempted, processing is allowed to continue. Otherwise, a security violation is indicated.

In the above example, user ENDU with access control class 1 through 8 is allowed to access sublibrary AUX.PR$302 due to the match on class 8. The access right is limited to UPDATE. An attempt by ENDU to ALTER (rename or delete) the sublibrary would be an access violation.

In case of an access violation, the job or user program is canceled, or execution of the function is skipped. The violation is recorded (*logged*) on the log data set if the optional program *VSE/Access Control-Logging and Reporting* is installed.

Access control classes are also used to determine whether **allowed** accesses to resources defined in DTSECTAB are to be logged. Please refer to the description of the LOG parameter on page 164.

In case there is more than one match between access control classes, the higher access right becomes effective.

Access Checking Flow

Figure 52 on page 155 shows the concept of access authorization checking (universal access rights and type 1 user access rights are not taken into account).

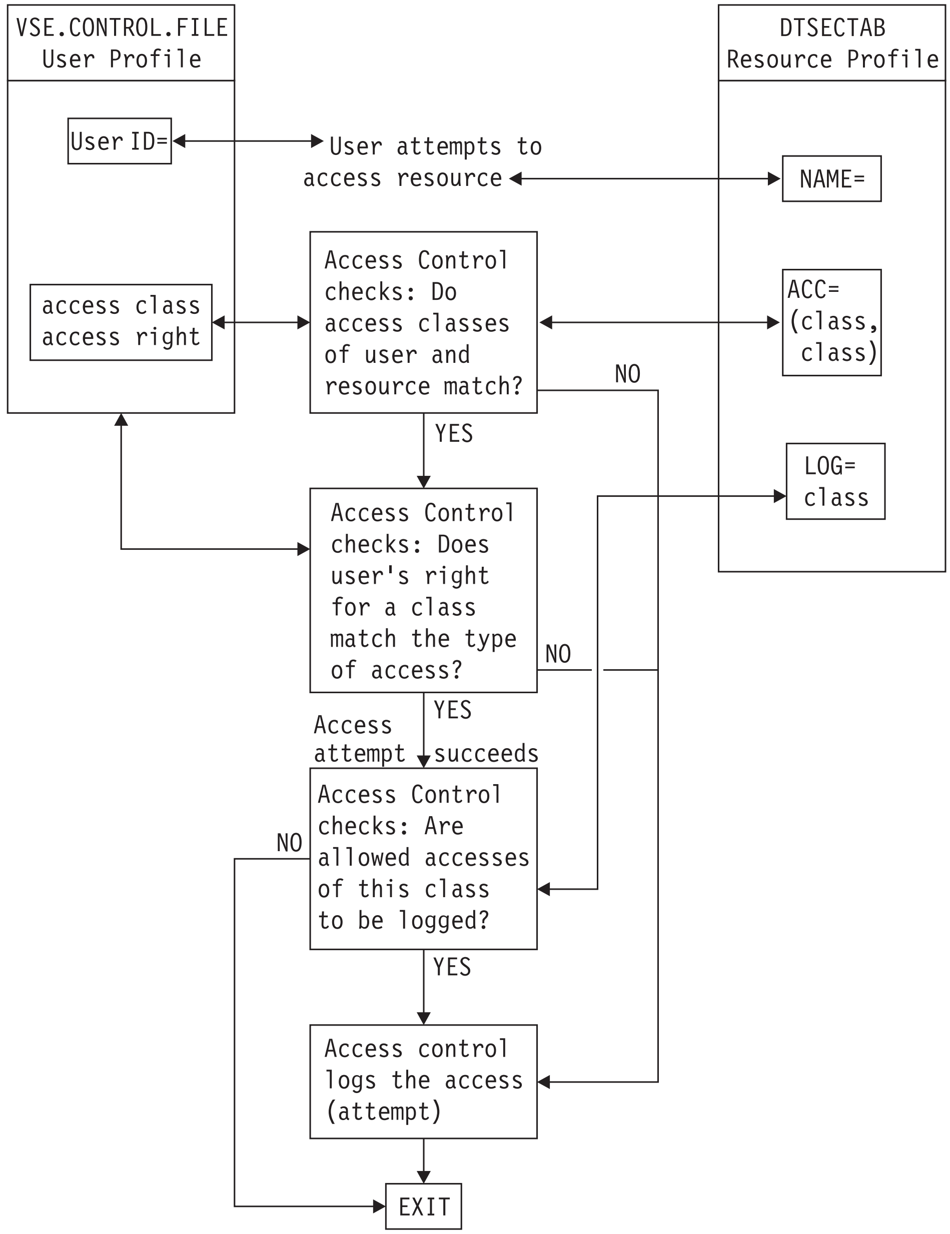


Figure 52. Access Authorization Checking with Access Control Classes Defined in DTSECTAB

Access Control for Libraries

The Access Control Function follows the hierarchy of the VSE library structure. If a library is protected, all sublibraries and members in it are protected automatically. If a library is **not** protected, its sublibraries and members **cannot** be protected.

The possession of any access right (except CON) to a library entity implies at least the same access right to the entities below it in the hierarchy. Consider this example:

```
Sublibrary REP1.DEV........... access right UPD
Member     REP1.DEV.PROG1...... access right READ
```

Member PROG1 automatically inherits the access right of UPD from sublibrary REP1.DEV. Its own specification of READ is overridden.

The Access Right of CON

The access right of CON is valid for libraries and sublibraries only. It determines whether a user may access the library/sublibrary at all, for example by a LIBDEF.

The access right of CON is **not** conferred to the next lower level. Therefore, in a library protected with CON, a sublibrary that has no entry for itself cannot be accessed.

Consider the following example:

```
Library PRD1 ............. access right CON

Sublibrary PRD1.BASE....... access right CON
Sublibrary PRD1.CONFIG..... no access right specified, or
                            not defined in DTSECTAB at all
```

The user can do a LIBDEF for sublibrary PRD1.BASE but **not** for sublibrary PRD1.CONFIG.

CON allows no higher access right such as READ or UPD. In particular, to be able to read or update members of a sublibrary that is protected with the access right of CON, you must provide profile entries in the access control table for all its members (be sure to use generic member notation). Otherwise, only the security administrator has access to these members.

Hierarchical Access Checking

For access rights other than CON, library protection is strictly hierarchical. If the user has an access right for a library, he has the same right for all sublibraries and all members therein.

When a protected resource in the library hierarchy is to be accessed, the access control checking is done at each level in the hierarchy, until a sufficient access right is found (see Figure 54 on page 181). For this reason,

- Access checking on the lower level is only done if the inherited access right from the higher level is not sufficient.
- Defining access rights at the lower level makes only sense if the lower level's right is higher than the right inherited from the higher level.

When a member is to be accessed, first the access right to the library is checked, then to the sublibrary. These rights are determined when the library and the sublibrary are accessed (for example by a LIBDEF). If their rights are not sufficient, the member itself will be checked. The user attempting the access must have at least the access right CON (connect) to the library and the sublibrary. If, for example, a user is attempting read access to the member, and has the READ access right to the **library** in which the member is stored, Access Control allows the access, and does no checking at sublibrary or member level.

The same is true if you attempt to access a protected sublibrary. In this case, you must at least have the access right CON (connect) to the library in which the sublibrary resides, and checking stops if you have a sufficient access right to that library.

Impact on Logging

Via the LOG parameter in the resource definition, you can specify for which classes successful accesses to a protected resource are to be logged. Access violations are always logged. An access granted via a universal access right is never logged.

The method of hierarchical access checking, as described before, has an impact on logging. The access to a member is only logged if access checking reaches the member level, that is: if there is no inheriting of a sufficient access right.

Consider the following example.

```
EXEC LIBR
ACCESS S=IJSYSRS.SYSLIB
LIST $IPLESA.PROC
```

- The access right for library IJSYSRS is established as the maximum of universal and (the job submitter's) individual access rights. This right (if larger than CON) is inherited by sublibrary IJSYSRS.SYSLIB.
- The access right for sublibrary IJSYSRS.SYSLIB is established as the maximum between the inherited right and the maximum of universal and individual access rights.

  This right (if larger than CON) is inherited by all members in IJSYSRS.SYSLIB.

For the

```
LIST $IPLESA.PROC
```

statement, an access right of READ is required.

- If the inherited access right is READ or higher, then no more access checking is done and access is allowed. The access is not logged.
- Else, the maximum of universal and individual access rights must be READ or higher to access the member, otherwise an access violation occurs.

  The access is logged if the UACC is not sufficient and the access class is specified in the LOG parameter.

Access Control for LIBDEF Statements

For certain functions (for example FETCH/LOAD, the linkage editor, or job control), access to sublibraries is requested by LIBDEF job control statements.

The system regards a LIBDEF statement as an attempt to access the sublibrary or sublibraries that it specifies. LIBDEF statements are of two types:

- Permanent (PERM), valid for all jobs in the partition in which they are entered;
- Temporary (TEMP), valid only within the job in which they are entered.

Access checking and the granting of access rights is done as described in the preceding sections.

Please note for **permanent** LIBDEFs:

For any sublibrary (except IJSYSRS.SYSLIB) which you intend to specify in a permanent LIBDEF statement, plus its containing library, you **must** specify a **universal access right** (UACC=CON or higher) in the resource profile.

Only the **universal** access right to the sublibrary is granted. This is because a permanent LIBDEF is still effective after completion of the job that established the LIBDEF. Another user's job can use this LIBDEF.

Please note for **temporary** LIBDEFs:

For a temporary LIBDEF statement, **individual** access rights of the user or **universal** access rights (whichever are higher) are granted.

When access to a sublibrary is via a **temporary** LIBDEF, the normal rules for checking apply. If the universal access right is sufficient, the access is allowed; if not, the individual access right is checked, and the access is allowed if this is sufficient.

This means, that a user who wants an individual access right to be granted for a sublibrary must use a temporary LIBDEF.

Access Checking for Source Library Inclusion (SLI)

Checking for the access right of READ (to a member) becomes necessary when a VSE/POWER job contains an * $$ SLI statement which includes a member from a **VSE library**.

There are two formats of * $$ SLI:

1. * $$ SLI MEM=
2. * $$ SLI MEM=... ,S=lib.sublib...

The first format specifies only the member name whereas the second format has also the **sublibrary** specified that contains the member.

Format 1 (with member name only): In this case, the VSE/POWER partition (via LIBDEF) must have the access right of at least CON for the member's library and sublibrary. If the VSE/POWER partition's access rights are not higher than CON, **the job** must have an access right of at least READ to the member.

In the z/VSE system that is shipped to you, the startup procedure for the VSE/POWER partition contains permanent LIBDEFs. Therefore only the universal access rights are established.

It is recommended that you retain the **permanent** LIBDEFs. Keeping these universal rights low allows you to carefully set the higher access rights on member level.

Format 2 (member plus sublibrary): In this case, no access checking against the VSE/POWER partition takes place. Rather, the job containing the SLI statement must provide the proper access right.

Special Access Checking for Librarian Commands

Normally, a user can access a member in a sublibrary if that user has the appropriate access right for the member and the connect (CON) right for the sublibrary.

However,

- Librarian commands with a **generic** member specification require the access right of READ (or higher) for the sublibrary, in addition to the appropriate access rights for the members. Alternatively, the access right of CON to a library or sublibrary and a user profile entry of READDIR=YES are sufficient for reading the respective directory.

- A Librarian TEST command specifying a sublibrary or member name always requires the access right of READ for the **library** in which the sublibrary or member resides.
- A Librarian SEARCH command with OUTPUT=FULL always requires the access right of READ for the **library** to be searched, even if only a sublibrary has been specified.

Protection of the System Library and System Sublibrary

The system library IJSYSRS and the system sublibrary IJSYSRS.SYSLIB are treated in a special way because they are accessed for the first time at IPL time when label information may not yet be available. Access Control needs label information for files and libraries.

When this label information is not available, a default must be taken. By default, the system **library**, IJSYSRS, has the universal access right of connect (UACC=CON) while the system **sublibrary**, IJSYSRS.SYSLIB, has the universal access right of read (UACC=READ).

The **default** also applies if the (sub)library has no entry in the access control table DTSECTAB. If the (sub)library has an entry in DTSECTAB which does not specifically defines a universal access right, the universal access right is set to CON.

Therefore, regardless of what you specify in your DTSECTAB, IJSYSRS and IJSYSRS.SYSLIB have at least a universal access right of connect (UACC=CON).

The access control table DTSECTAB that is delivered with your z/VSE system defines universal access rights of connect (UACC=CON) for all system sublibraries. This allows to exercise selective control over certain programs (for example DITTO/ESA for VSE in PRD1.BASE).

Protection of PRIMARY Library and Sublibraries

During initial installation, a VSE library named PRIMARY is automatically created. This library will later contain PRIMARY.userid sublibraries for all users defined in z/VSE's control file.

PRIMARY sublibraries allow for **private user libraries** that offer a similar kind of protection as the *primary libraries* in a VSE/ICCF environment.

Accessing PRIMARY Sublibraries: The name of such a sublibrary is always PRIMARY.userid.

No entries in the access control table DTSECTAB are required for PRIMARY **sublibraries**. An entry for the PRIMARY library is sufficient and is provided in the pregenerated table of your z/VSE system.

Without explicit authorization in the access control table DTSECTAB, only the owner user ID and a type 1 user (system administrator) can access the data stored in such a sublibrary. Note that such a user can access all resources.

The owning user automatically has the UPDATE right. The access right can be increased to ALTER by appropriate entries in DTSECTAB. It is not possible to reduce in DTSECTAB a user's UPDATE right to a sublibrary owned by this user.

For a type 2 user (programmer) and a type 3 user (operator), Access Control checks whether the user ID of the requestor and the name of the PRIMARY sublibrary match. If user ID and name do not match, no access is possible without appropriate DTSECTAB entries.

PRIMARY Sublibraries for Predefined Users SYSA, OPER and PROG: The above users have by default a primary sublibrary when the system is newly installed.

PRIMARY.$$C Common Sublibrary: A special sublibrary, named PRIMARY.$$C, is also available to allow data exchange among different users. Sublibrary PRIMARY.$$C is protected in the pregenerated DTSECTAB with the universal access right of update (UPD). Therefore, all users have read/write access to this sublibrary.

Access Control for Startup Procedures

Startup procedures, such as $0JCL, include // ID statements for user FORSEC for accessing protected resources. The statements are processed if the system has security on SEC=YES. They are ignored if SEC=NO is specified. The password is not necessary during system startup and is therefore omitted from the // ID statement.

For each partition, the startup procedure is loaded without access checking. Neither a universal nor an individual access right is necessary.

Startup Procedures with Access Rights of a Particular User

The following applies to startup processing for static partitions.

1. If the `// ID` statement in an ASI procedure contains no password, then password verification is skipped.
2. All access checks from now on till end-of-job (/&) are done using the information from the specified user's profile.
3. Batch jobs that are submitted to VSE/POWER inherit the user ID for which startup is running. If desired, this user ID propagation can be overridden by including a // ID statement with another user ID and the respective password in the job streams or by coding this user information in the * $$ JOB statement of the submitted jobs.

Access Control and CICS Region Prefix

In an environment with more than one CICS region, the CICS prefixing allows an installation to prevent users on one CICS region to access resources of a different CICS region that has a different prefix.

The BSM supports the CICS prefixing of transaction names (see "Protecting CICS Transactions with Access Control Table DTSECTXN" on page 135).

CICS uses the user ID under which the CICS region is running as prefix. If the system is started with SYS SEC=YES, this user ID is the ID of the job submitter, the POWER job card, or the // ID statement. This may result in short (4 character) prefixes which are normally not very meaningful or it exposes the password. To avoid this, special task user IDs are provided by BSM. They should be specified in the // ID statement without password. Jobs, which have such a user ID in the ID statement have to be submitted from an administrator (type 1 user ID). IBM ships two special task user IDs (initial name is DUMMY), DBDCCICS and PRODCICS to be used for CICS startup. No logon to Interactive Interface is possible with these user IDs.

If no DTSECTAB security is active (SYS SEC=NO), the BSM takes the user ID from the // ID statement in the CICS startup job. This // ID statement does not require a password. If no // ID statement is found, the user ID FORSEC is used as default.

System Phases, B-Transients, Link Area, SVA and LTA

In the pregenerated access control table DTSECTAB, the entries

```
DTSECTAB TYPE=SUBLIB,                                              C
      NAME=IJSYSRS.SYSLIB,                                         C
      UACC=CON
DTSECTAB TYPE=MEMBER,                                              C
      NAME=IJSYSRS.SYSLIB.*,                                       C
      UACC=READ
```

ensure that system phases can be executed at any time and for any user. If for some reason you lower the UACC in the second call (TYPE=MEMBER), you should increase the UACC in the first call (TYPE=SUBLIB) to at least READ.

Considerations for B-Transients

B-Transients are a special kind of system phases. In a system with security active, they can be loaded from protected libraries only. The attempt to load a B-transient from an unprotected library would cause an access violation.

Likewise, **in a system with security active, B-transients can be cataloged only in protected libraries**.

Libraries that contain B-transients of the z/VSE **base programs** are automatically protected by appropriate entries in the pregenerated access control table DTSECTAB.

Considerations for Link Area, SVA, and LTA

The Link Area (used when link-editing with OPTION LINK), the Shared Virtual Area (SVA), and the Logical Transient Area (LTA) cannot be protected explicitly by entries in DTSECTAB.

The **Link Area** is considered as an unprotected library. This means that if OPTION LINK is used, and a name beginning with $$B... appears in the PHASE statement, an access violation occurs.

For the execution of phases, the **SVA** is regarded as a resource with the universal access right READ.

The **LTA** is activated and used by special phases, the B-transients. The rules for B-transients apply (see the preceding section). B-transients which do not conform to these rules cannot be activated, and cause an access violation when called.

Defining User Entries in the VSE.CONTROL.FILE

User profiles are stored in the VSE.CONTROL.FILE (and not in DTSECTAB) *except for* the predefined users FORSEC and DUMMY. These predefined users are required by z/VSE, and are the only user definitions included in DTSECTAB. User FORSEC is defined in both DTSECTAB and VSE.CONTROL.FILE. **Do not delete these users!**

You can use the *Maintain User Profiles* dialog to create or maintain user profiles.

Using the Maintain User Profiles Dialog

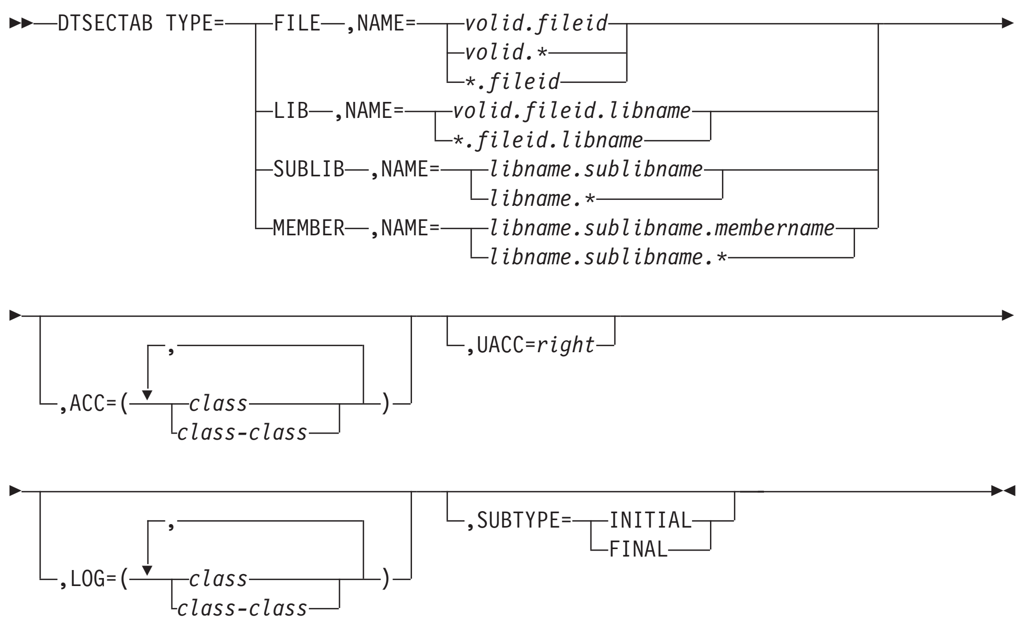
The Interactive Interface provides the *Maintain User Profiles* dialog you can use to define, update, or delete user profiles. For a detailed description of this dialog, refer to "Maintaining User Profiles" on page 98.

Defining Resource Entries in DTSECTAB

The DTSECTAB macro is used to build the access control table DTSECTAB and to define resource entries in it. You must specify one entry for each resource to be protected. If generic specification of resource names is used, an arbitrary number of resources can be protected with one table entry.

For each entry to be specified a DTSECTAB macro call is required. The first macro call must include the SUBTYPE=INITIAL parameter, the last one the SUBTYPE=FINAL parameter.

Format of DTSECTAB Macro for Defining Resources



Mandatory Parameters:

TYPE Specifies for which type of resource this profile is intended. The possible values are:

FILE	for a file profile,
LIB	for a library profile,
SUBLIB	for a sublibrary profile
MEMBER	for a member profile.

NAME
Specifies the name of the resource. The format of the value must correspond to the value of the TYPE parameter, as follows:

TYPE	**Format of NAME Value**
FILE	volid.fileid

LIB volid.fileid.libname

SUBLIB
libname.sublibname

MEMBER
libname.sublibname.membername

The elements of the resource names are:

volid The 1 to 6-character volume identifier of the volume on which the file or library resides. For files on tape, for files or libraries in VSE/VSAM-managed space, and for VSE/VSAM files code an asterisk (*) for **volid**.

fileid The 1 to 44-character fileid (the second operand of the DLBL statement) or the 17-byte tape-fileid (TLBL) as defined for the file or library.

libname
The library name used in the librarian DEFINE command which defined the library.

sublibname
The sublibrary name used in the librarian DEFINE command which defined the sublibrary.

membername
The member name under which the member was cataloged. The member type is not used in the access control table. Members of all types having the same name share the same access control profile.

Notes:

1. A library is, in practice, a file which has been put under the control of the librarian program using a librarian DEFINE command. Be careful not to submit a FILE-type macro in addition to the LIB-type macro for a library.
2. "libname" must be used only once in the system, so that the NAME parameters for sublibrary entries can be made unique.
3. The combination "volid.fileid" must be used only once in the system.
4. The "file-id" must be identical to the file-id in the DLBL or TLBL statement. It may contain periods; the access control function does not confuse these with the periods delimiting the file-id from the other operands.
5. Specification and padding of "volid" and "file-id" are as described for the "DLBL", "TLBL" and "EXTENT". statements in the IBM manual *z/VSE System Control Statements*.
6. For a multi-extent, multi-volume library residing in non-VSAM space, only one entry with TYPE=LIB is needed. The VOLID must be of the volume on which the first extent of the library resides.

Optional Parameters:

ACC Specifies the access control class(es) to be assigned to the named resource. For **class**, specify a decimal number from 1 to 32. You can also specify a list of classes, for example (1,3,5), or a range of classes, for example (1-3). Access is allowed to those users who have one or more of the specified classes in the ACC parameter of their user profile.

UACC For **libraries, sublibraries and members only:** Specifies an access right which **all** users of the system are to have to the named resource. This right is granted irrespective of the classes specified in the users' profiles. The possible values for **right** are:

ALT, UPD, READ and CON.

See Table 3 on page 152 for explanations of the access rights.

LOG Specifies which class(es) of **successful accesses** are to be logged. Access **violations** are always logged.

The same syntax as for the ACC parameter applies. Only classes that appear in the ACC parameter are meaningful here.

SUBTYPE
This parameter is required in the first and the last call of the DTSECTAB macro, and is not allowed in the intermediate calls. The value **INITIAL** must be used in the first call, and the value **FINAL** in the last call.

Generic Protection of Resources

The rules for single resources, as explained above, also apply to groups of resources. In the DTSECTAB macro, you can make a **generic** specification in the NAME parameter. To do this, code an asterisk (*) for the **last** element of the resource name. For example, ...NAME=000111.*... would apply the access control profile to **all files** on the volume with the serial number 000111. Coding ...NAME=LIB1.SUBA.*... would apply the profile to **all members** in sublibrary SUBA of library LIB1.

Note that the asterisk (*) is a reserved character and must not be used in resource names. Therefore, do not use the asterisk in a file name or in a volume serial number.

To apply an access control profile to, for example, all files on volume 000111 whose file-ids begin with PAY..., code:

```
NAME=000111.PAY*
```

The common part of the resource names can be any length, up to one character less than the allowed maximum length. In other words, the generic name, including the asterisk, is subject to the same length restrictions as a normal resource name.

Note: You can give a group of members the same access class(es) by generic specification in the DTSECTAB macro. However, this is not sufficient to access these members with a generic specification in a Librarian command. You still need the access right of READ or higher to the sublibrary or, for reading directory information, the access right of CON plus READDIR=YES in the user profile (see "Special Access Checking for Librarian Commands" on page 158).

Generic protection is not available at library level.

A special case arises when there are two or more generic specifications starting with the same characters. For example, two entries of type FILE might have the parameters:

```
NAME=000999.PAY*,ACC=1
```

and

```
NAME=000999.PAYR*,ACC=2
```

If an access to the file PAYROLL on volume 000999 is attempted, the Access Control Function will check for (and allow) access class 2 and reject access class 1. The rule is that the longest generic name which matches with the name of the accessed resource is used.

If the full name of the resource to be accessed (in this example, PAYROLL) is found in an entry, this entry is, of course, used for checking. Or take as another example the two specifications

```
IJSYSRS.SYSLIB.*,UACC=READ
IJSYSRS.SYSLIB.DITTO
```

All members of the SYSLIB sublibrary have the universal access right of READ, but the program DITTO is excluded from this rule.

Examples of DTSECTAB Resource Entries

File Entries:

```
DTSECTAB TYPE=FILE,            X
      NAME=999999.PAYROLL,     X
      ACC=(16),                X
      LOG=(16)
```

This entry can be used to protect non-VSAM disk files. The protected name consists of the six-character volume-ID of the storage medium used, followed by the 44-byte file-id, taken from the DLBL statement.

```
DTSECTAB TYPE=FILE,            X
      NAME=*.PAYROLL,          X
      ACC=(16),                X
      LOG=(16)
```

This entry can be used to protect tape files or SAM files in VSE/VSAM-managed space. The protected resource name must consist of an asterisk (*) followed by the 44-byte (disk) or 17-byte (tape) file-id, taken from the DLBL or TLBL statement, respectively.

In both examples, only users with the access class 16 or AUTH=YES in their access control profile can access the file PAYROLL. Successful accesses will be logged (LOG parameter).

Library Entries:

```
DTSECTAB TYPE=LIB,             X
      NAME=888888.P.C.TEST.LIB1, X
      ACC=(1-8,32),            X
      LOG=(1-8)
```

Library entries are used for protecting libraries. Protecting a library means also protecting the sublibraries and members in it. The name consists of the 6-character volume-ID of the storage medium used, followed by the file-id from the DLBL statement defining the disk file, and the filename from the current DLBL statement associated with the library.

Users with access classes in the range 1 to 8 and 32 can access this library (ACC parameter), and all entities in it. Accesses made via classes 1 to 8 will be logged.

It is possible to allow users who need to access only some of the sublibraries in this library to do so. This is done in the following example by using the universal access right connect (UACC=CON):

```
DTSECTAB TYPE=LIB,              X
      NAME=888888.TEST.LIB1,    X
      ACC=(1-8,32),             X
      UACC=CON,                 X
      LOG=(1-8)
```

If a sublibrary in this library is given, for example, the access class 10, a user with only access class 10 in his profile could access that sublibrary (but no other sublibrary in this library).

Note: The examples above are for libraries in non-VSAM space. For libraries in VSE/VSAM-managed space, code an asterisk (*) in place of the volume identifier.

Sublibrary Entries:

```
DTSECTAB TYPE=SUBLIB,           X
      NAME=LIB1.SUBA,           X
      ACC=(4-12),               X
      LOG=(4-12)
```

Sublibrary entries are used for defining access rights to sublibraries. The name consists of the name of the library, followed by the name of the sublibrary. The sublibrary name is the name used in the librarian DEFINE command which created the sublibrary.

Users with access classes in the range 4 to 12 can access this sublibrary (ACC parameter), and all members in it. Accesses made via classes 4 to 12 will be logged.

It is possible to allow users who need to access only some of the members in this sublibrary to do so. This is done in this example by using the universal access right connect (UACC=CON):

```
DTSECTAB TYPE=SUBLIB,            X
      NAME=LIB1.SUBA,            X
      ACC=(4-12),                X
      UACC=CON,                  X
      LOG=(4-12)
```

If a member is given the access class, for example, 3, a user with only access class 3 in the user profile could access that member.

Member Entries:

```
DTSECTAB TYPE=MEMBER,             X
      NAME=LIB1.SUBA.PAY*,        X
      ACC=(9-15),                 X
      LOG=(9-10,15)

DTSECTAB TYPE=MEMBER,             X
      NAME=LIB1.SUBX.DITTO,       X
      SUBTYPE=FINAL
```

Member entries define access rights to members in sublibraries.

The value in the NAME parameter consists of the library name, sublibrary name and member name. The member type is not specified. All types of member (phases, object modules, procedures and so on) with the same name have the same access profile. This means, for example, that if a programmer has the “alter” access right to the source book of a program, he will automatically have the same right to the object module and the phase form of the program. The programmer can also catalog a procedure of the same name to set up the job control statements for the program. The first example has a generic name specification. This means that the access profile is the same for all source books, object modules, phases and procedures in this sublibrary whose names begin with PAY.

Note that generic protection does not necessarily allow generic access to members in Librarian commands (see “Generic Protection of Resources” on page 164).

SUBTYPE=FINAL, as in the second example, must be specified for the last macro call in the assembler input stream.

Example of DTSECTAB Entries for Library Control

The following DTSECTAB entries demonstrate the various protection levels available for controlling the access to libraries, sublibraries, and their members. The example is based on sublibrary REP1.BASE, which is located in VSE/VSAM-managed space. This sublibrary is assumed to have stored a phase with the name PROG1.

Four DTSECTAB entries are shown followed by explanatory text.

1. Protect the library:

```
DTSECTAB TYPE=LIB,                                 X
         NAME=*.VSE.REP1.LIBRARY.REP1,             X
         UACC=CON
```

2. Protect the sublibrary:

```
DTSECTAB TYPE=SUBLIB,                              X
         NAME=REP1.BASE,                           X
         UACC=CON
```

3. Allow each user to read all members:

```
DTSECTAB TYPE=MEMBER,                              X
         NAME=REP1.BASE.*,                         X
         UACC=READ
```

4. Restrict access to PROG1 to the security administrator:

```
DTSECTAB TYPE=MEMBER,                              X
         NAME=REP1.BASE.PROG1                      X
```

The **first two** entries protect sublibrary REP1.BASE and give all users the access right of CON. (By substituting or adding to UACC=CON in the entry for the sublibrary an ACC= statement, you could give selected users higher access rights to REP1.BASE).

Entry **three** allows all users read access to all members in sublibrary REP1.BASE.

Entry **four** restricts access to PROG1 to the security administrator. The member is protected against any access from other users because neither a universal nor an individual access right is defined.

Entry **four** could be expanded to look like this:

```
DTSECTAB TYPE=MEMBER,                              X
         NAME=REP1.BASE.PROG1,                     X
         ACC=(10,12)
```

This restricts access to PROG1 to selected users (other than a security administrator). These users must have access control classes 10 or 12 defined in their user profile.

Predefined DTSECTAB Security Support

As shipped, z/VSE provides basic security support for access control which is ready to use. This support includes the following functions and resources:

- Activation of the Basic Security Manager (BSM) during initial installation.
- Startup procedures adapted for a system with the Basic Security Manager active.
- A predefined access control table (DTSECTAB) for resource protection which is automatically generated during initial installation, if you answered with YES during initial installation .
- A predefined access control table (DTSECTXN) for protecting CICS transactions is provided by z/VSE. DTSECTXN is not discussed in this section, but under "Protecting CICS Transactions with Access Control Table DTSECTXN" on page 135.
- PRIMARY sublibraries for predefined users SYSA, OPER, and PROG.

Activating the Predefined DTSECTAB Security Support

You can activate basic security for batch resources either during initial installation, or later by using the *Tailor IPL Procedure* dialog as described under "Using the Tailor IPL Procedure Dialog to Modify Security Parameters" on page 133.

If the support is activated **during initial installation**, the installation program updates the IPL procedure $IPLESA and adds the statement:

```
SYS SEC=(YES,NOTAPE)
```

NOTAPE means that files on disk are protected but not files on tape. SEC=YES activates both, the protection of disk and the protection of tape files.

In a system with security on, the predefined access control table DTSECTAB becomes active at the first IPL after initial installation.

This ensures a correct startup for a system with security active. You should use the predefined DTSECTAB as a base when later adding your own entries. The table provides the necessary protection of system libraries. Startup would not work if system libraries were not properly protected.

Tasks to be Done after Initial Installation

During initial installation the system defines the users shown in the table in Table 1 on page 1.

The first IPL after initial installation with SEC=YES activates the predefined basic security support. A system administrator (type 1 user, SYSA) has unlimited access to all resources and should now do the following:

- Change the passwords of certain user IDs as described on page 1.
- For users PROG, OPER and $SRV log-on directly to change the password. Afterwards, use the *Maintain Primary Sublibraries* dialog to define PRIMARY sublibraries for the users SYSA, PROG, and OPER. The password for user POST needs to be changed after the initial installation process has been completed. Change the password via the dialog *Maintain User Profiles*, (for details refer to the section "Maintaining User Profiles" on page 98).

- Change and submit skeleton SKCICS and possibly SKCICS2 in VSE/ICCF library 59. Log on as system administrator, and submit the job. This submission catalogs the startup job CICSICCF as member CICSICCF.Z in the system sublibrary IJSYSRS.SYSLIB and places the job into the VSE/POWER reader queue. The job ″inherits″ the security attributes of the submitter and changes them via the ID statement to the special task userID for the CICS region. Make sure this is the only CICS startup job by deleting the original from the reader queue.

As PAUSExx jobs do not have any access rights, you can provide access rights via the // ID statement if required.

Considerations for User IDs FORSEC and DUMMY

Users FORSEC and DUMMY are available for system purposes only. They are the only users defined in DTSECTAB.

FORSEC

is defined as **system administrator** in the z/VSE system that is delivered to you. Its purpose is to provide appropriate access rights **at system startup**.

Important

Never remove user FORSEC from neither DTSECTAB nor the VSE.CONTROL.FILE.

If your installation uses the optional program *VSE/Access Control - Logging and Reporting (VSE/ACLR)*, you may want to reduce the access rights of this user by specifying access control **classes.** This greatly reduces the number of logging records because **administrator** accesses are always logged (not only violations). Therefore, activate the Logging and Reporting program only after you have reduced the access rights of user FORSEC in DTSECTAB and the VSE control file. Refer also to “Logging and Reporting Accesses to DTSECTAB Resources” on page 178.

The manual *VSE/Access Control-Logging and Reporting: Program Reference and Operations Guide* provides details about VSE/Access Control - Logging and Reporting.

DUMMY

has no special access rights. This user ID is included in certain startup procedures before certain jobs (such as PAUSEBG, PAUSEF1, ...) are submitted. User ID DUMMY serves to inhibit the inheritance of the user FORSEC’s access rights (which are administrator rights) to those jobs.

Important

Never remove user DUMMY from DTSECTAB.

Pregenerated Access Control Table DTSECTAB

As mentioned before, the z/VSE system that is delivered to you has a predefined access control table DTSECTAB. The table provides a basic level of security; it does not use any access control classes. It is automatically generated and ready for use immediately after initial installation.

You should **never remove or change** any of the supplied entries without careful consideration. You may, of course, add entries by using the methods described below.

Predefined Security Table DTSECTRC

DTSECTRC has security information about protected **z/VSE system resources** (plus definitions of special users: FORSEC and DUMMY). Its content is shown under "Content of Pregenerated DTSECTAB (DTSECTRC in VSE/ICCF Library 59)" on page 171. (The exact content may have changed since this manual was printed. Please look at your copy in VSE/ICCF library 59, member DTSECTRC.)

You may add entries for protected z/VSE system resources as needed or change those provided by adding access control classes. Programs that can potentially bypass access control, should also be protected at member level. Be sure that no person other than the security administrator is given an access right to sensitive members.

Note: Except for users FORSEC and DUMMY, users should only be defined in VSE.CONTROL.FILE (user SYSA).

Section "Maintaining the Access Control Table DTSECTAB" on page 176 describes how to maintain DTSECTAB.

Content of Pregenerated DTSECTAB (DTSECTRC in VSE/ICCF Library 59)

```
         TITLE 'DTSECTAB - SECURITY TABLE FOR RESOURCES'
*********************************************************************
         PUNCH ' CATALOG DTSECTRC.OBJ  REP=YES'
         SPACE 3
*-------------------------------------------------------------------*
*                                                                   *
*       STATIC PART OF DTSECTAB                                     *
*                                                                   *
*-------------------------------------------------------------------*
*       THIS PART IS SHIPPED AS A-BOOK IN IJSYSRS.SYSLIB.DTSECTRC.   *
*       IF CHANGED, THE USER SHOULD PUT HIS VERSION UNDER THE SAME   *
*       NAME IN PRD2.SAVE, AS IBM SERVICE IS DONE ON THE MEMBERS     *
*       CONTAINED IN IJSYSRS.SYSLIB.                                 *
*       (THE JOB TO BUILD A DTSECTAB LOOKS FIRST IN PRD2.SAVE FOR    *
*       DTSECTRC).                                                  *
*-------------------------------------------------------------------*
*-------------------------------------------------------------------*
*                IBM SUPPLIED USERS                                 *
*-------------------------------------------------------------------*
*** USER DUMMY HAS NO SPECIAL SECURITY RIGHTS.USED TO RESET INHERITANCE
*** IT AVOIDS GETTING TOO MANY RIGHTS WHILE LOADING POWER JOBS DURING
*** ASI.
*** YOU SHOULD NOT DEFINE AN II USER WITH THE NAME 'DUMMY'.
*-------------------------------------------------------------------*
         DTSECTAB TYPE=USER,                                        C
               NAME=DUMMY,                                          C
               PASSWRD=DUMMY,                                       C
               AUTH=NO,                                             C
               SUBTYPE=INITIAL
         SPACE 3
*-------------------------------------------------------------------*
*** USER FORSEC HAS ALL ACCESS RIGHTS. THEREFORE, THE PASSWORD NEEDS
*** TO BE CHANGED AFTER INITIAL INSTALLATION.
*-------------------------------------------------------------------*
         DTSECTAB TYPE=USER,                                        C
               NAME=FORSEC,                                         C
               PASSWRD=FORSEC,                                      C
               READDIR=YES,                                         C
               MCONS=YES,                                           C
               AUTH=YES,                                            C
               RIGHT=BTRANS
*-------------------------------------------------------------------*
*                END OF IBM SUPPLIED USERS                          *
*-------------------------------------------------------------------*
```

Figure 53. Pregenerated DTSECTAB (DTSECTRC in VSE/ICCF Library 59) (Part 1 of 5)

```
*-------------------------------------------------------------------*
*                                                                   *
*       FOLLOWING IS THE Z/VSE 3.1 SUPPLIED PART OF THE DTSECTAB    *
*       THAT DEFINES A MINIMUM SET OF RESOURCES TO BE PROTECTED.    *
*                                                                   *
*-------------------------------------------------------------------*
****** LIBRARIES*****************************************************
******   IJSYSRS
         DTSECTAB TYPE=LIB,                                         C
               NAME=DOSRES.VSE.SYSRES.LIBRARY.IJSYSRS,              C
               UACC=CON
         DTSECTAB TYPE=SUBLIB,                                      C
               NAME=IJSYSRS.SYSLIB,                                 C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.*,                               C
               UACC=READ
******   CPUVAR* IS USED BY VARIOUS JOBSTREAMS TO SAVE PARAMETERS
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.CPUVAR*,                         C
               UACC=UPD
******   ALLOW PROGRAMMER TO ADD HIS/HER OWN VSAM FILE VIA II DIALOGS
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.STDLABUP,                        C
               UACC=UPD
******   CLRDK DESTROYS THE DATA ON A DISK.
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.CLRDK
******   ICKDSF DESTROYS DATA ON A DISK.
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.ICKDSF
******   IKQVDU CHANGES THE FORMAT 1 LABEL
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.IKQVDU
******   DTSANALS SHOULD BE EXECUTED BY AUTHORIZED PERSONS ONLY
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.DTSANALS
******   DTSUTILA IS A RESOURCE THAT PROTECTS SECURITY SENSITIVE
*        DTSUTIL COMMANDS FROM BEING EXECUTED BY NON-ADMINISTRATORS
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.DTSUTILA
******   SECURITY PROGRAMS/TABLES
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.DTSEC*
******   PROGRAMS TO MANIPULATE THE CONTROL FILE
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.IESUPDCF
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.IESBLDUP
******   PROGRAMS TO ALLOW POWER Q MANIPULATION
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.DTRIJMGR
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.IPW$$DD
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.BSTADMIN
******   THE NEXT ENTRIES ARE USED TO PROTECT THE PASSWORDS IN ...
*        (PROGRAMS THAT ACCES THE VSE.CONTROL.FILE)
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.CICSICCF
         DTSECTAB TYPE=MEMBER,                                      C
               NAME=IJSYSRS.SYSLIB.CICS2
```

Figure 53. Pregenerated DTSECTAB (DTSECTRC in VSE/ICCF Library 59) (Part 2 of 5)

```
******   IJSYSR2 IS ALIAS NAME OF IJSYSRS, USED BY SERVICE DIALOGS
*        WHICH CAN ONLY INVOKED BY SYSTEM ADMINISTRATOR
         DTSECTAB TYPE=LIB,                                          C
               NAME=DOSRES.VSE.SYSRES.LIBRARY.IJSYSR2,               C
               UACC=CON
******   IJSYSR1 IS ALIAS NAME OF IJSYSRS, USED BY SERVICE DIALOGS
*        WHICH CAN ONLY INVOKED BY SYSTEM ADMINISTRATOR
         DTSECTAB TYPE=LIB,                                          C
               NAME=SYSWK1.SYS.NEW.RES.IJSYSR1,                      C
               UACC=CON
******   PRD1
         DTSECTAB TYPE=LIB,                                          C
               NAME=*.VSE.PRD1.LIBRARY.PRD1,                         C
               UACC=CON
******   PRD1.BASE
         DTSECTAB TYPE=SUBLIB,                                       C
               NAME=PRD1.BASE,                                       C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                       C
               NAME=PRD1.BASE.*,                                     C
               UACC=READ
******   PRD1.BASED SERVICE SUBLIBRARY
         DTSECTAB TYPE=SUBLIB,                                       C
               NAME=PRD1.BASED,                                      C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                       C
               NAME=PRD1.BASED.*,                                    C
               UACC=READ
******   PRD1.MACLIB
         DTSECTAB TYPE=SUBLIB,                                       C
               NAME=PRD1.MACLIB,                                     C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                       C
               NAME=PRD1.MACLIB.*,                                   C
               UACC=READ
******   PRD1.MACLIBD SERVICE SUBLIBRARY
         DTSECTAB TYPE=SUBLIB,                                       C
               NAME=PRD1.MACLIBD,                                    C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                       C
               NAME=PRD1.MACLIBD.*,                                  C
               UACC=READ
******   AVOID THAT ANYONE CAN MANIPULATE FILES USING DITTO
         DTSECTAB TYPE=MEMBER,                                       C
               NAME=PRD1.BASE.DITTO
******   PRDPRIM IS ALIAS NAME OF PRD1, USED BY SERVICE DIALOGS
*        WHICH CAN ONLY BE INVOKED BY SYSTEM ADMINISTRATOR
         DTSECTAB TYPE=LIB,                                          C
               NAME=*.VSE.PRD1.LIBRARY.PRDPRIM,                      C
               UACC=CON
******   PRD2
         DTSECTAB TYPE=LIB,                                          C
               NAME=*.VSE.PRD2.LIBRARY.PRD2,                         C
               UACC=CON
         DTSECTAB TYPE=SUBLIB,                                       C
               NAME=PRD2.*,                                          C
               UACC=READ
```

Figure 53. Pregenerated DTSECTAB (DTSECTRC in VSE/ICCF Library 59) (Part 3 of 5)

```
******   PRD2.SCEEBASE LE CODE LIBRARY
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.SCEEBASE,                                     C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.SCEEBASE.*,                                   C
               UACC=READ
******   PRD2.SCEEBASD LE SERVICE LIBRARY
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.SCEEBASD,                                     C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.SCEEBASD.*,                                   C
               UACC=READ
******   PRD2.DBASE PRODUCT LIBRARY DATABASES
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.DBASE,                                        C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.DBASE.*,                                      C
               UACC=READ
******   PRD2.PROD PRODUCT LIBRARY IN GENERAL
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.PROD,                                         C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.PROD.*,                                       C
               UACC=READ
******   PRD2.COMM PRODUCT LIBRARY COMMUNICATION PRODUCTS 1
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.COMM,                                         C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.COMM.*,                                       C
               UACC=READ
******   PRD2.COMM2 PRODUCT LIBRARY COMMUNICATION PRODUCTS 2
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.COMM2,                                        C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.COMM2.*,                                      C
               UACC=READ
******   PRD2.AFP PRODUCT LIBRARY ADVANCED PRINTER
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.AFP,                                          C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.AFP.*,                                        C
               UACC=READ
******   PRD2.DB2720 PRODUCT LIBRARY DB2 7.2.0
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.DB2720,                                       C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.DB2720.*,                                     C
               UACC=READ
******   PRD2.DB2STP PRODUCT LIBRARY DB2 STORED PROCEDURES
         DTSECTAB TYPE=SUBLIB,                                         C
               NAME=PRD2.DB2STP,                                       C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                         C
               NAME=PRD2.DB2STP.*,                                     C
               UACC=READ
```

Figure 53. Pregenerated DTSECTAB (DTSECTRC in VSE/ICCF Library 59) (Part 4 of 5)

```
******   SRV$SYS IS USED BY BACKUP JOBSTREAMS TO SAVE PARAMETERS
         DTSECTAB TYPE=MEMBER,                                          C
               NAME=PRD2.CONFIG.SRV$SYS,                                C
               UACC=UPD
******   BASIC START NEEDS THIS SUBLIB
         DTSECTAB TYPE=SUBLIB,                                          C
               NAME=PRD2.SAVE,                                          C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                          C
               NAME=PRD2.SAVE.*,                                        C
               UACC=UPD
******   ONLY THE SA IS ALLOWED TO READ THE SECURITY RELATED MEMBERS
         DTSECTAB TYPE=MEMBER,                                          C
               NAME=PRD2.SAVE.DTSEC*
         DTSECTAB TYPE=MEMBER,                                          C
               NAME=PRD2.SAVE.DTRI*
         DTSECTAB TYPE=SUBLIB,                                          C
               NAME=PRD2.CONFIG,                                        C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                          C
               NAME=PRD2.CONFIG.*,                                      C
               UACC=UPD
******   ONLY THE SA IS ALLOWED TO READ THE SECURITY RELATED MEMBERS
         DTSECTAB TYPE=MEMBER,                                          C
               NAME=PRD2.CONFIG.DTSEC*
         DTSECTAB TYPE=SUBLIB,                                          C
               NAME=PRD2.BSXCPU*,                                       C
               UACC=ALT
******   PRIMARY LIBRARY
         DTSECTAB TYPE=LIB,                                             C
               NAME=*.VSE.PRIMARY.LIBRARY.PRIMARY,                      C
               UACC=CON
******   THE $$C SUBLIB SHOULD BE USED TO EXCHANGE DATA BETWEEN USERS
         DTSECTAB TYPE=SUBLIB,                                          C
               NAME=PRIMARY.$$C,                                        C
               UACC=UPD
******   CRYPTO LIBRARY SSL KEYS
         DTSECTAB TYPE=LIB,                                             C
               NAME=*.VSE.CRYPTO.LIBRARY.CRYPTO,                        C
               UACC=CON
******   CRYPTO.KEYRING
         DTSECTAB TYPE=SUBLIB,                                          C
               NAME=CRYPTO.KEYRING,                                     C
               UACC=CON
         DTSECTAB TYPE=MEMBER,                                          C
               NAME=CRYPTO.KEYRING.*
******  FILES ****************************************************************
        DTSECTAB TYPE=FILE,                                             C
               NAME=*.VSE.CONTROL.FILE
        DTSECTAB TYPE=FILE,                                             C
               NAME=*.VSE.BSTCNTL.FILE
        DTSECTAB TYPE=FILE,                                             C
               NAME=DOSRES.VSE.POWER.QUEUE.FILE
        DTSECTAB TYPE=FILE,                                             C
               NAME=SYSWK1.VSE.POWER.DATA.FILE,                         C
               SUBTYPE=FINAL
         EJECT
*----------------------------------------------------------------------*
*                 END OF Z/VSE DTSECTAB                                *
*----------------------------------------------------------------------*
```

Figure 53. Pregenerated DTSECTAB (DTSECTRC in VSE/ICCF Library 59) (Part 5 of 5)

Maintaining the Access Control Table DTSECTAB

As discussed earlier, VSE Access Control offers two major groups of access rights for resources:

- **Universal** access rights via the UACC parameter, and/or
- **Individual** access rights via the ACC parameter.

Three scenarios will be discussed in the following:

1. Only the pregenerated security support is used: you do not add **resources** of your own to DTSECTAB, just **users** in the VSE.CONTROL.FILE.
2. You add users plus resources of your own. The resources are protected with universal access rights only. Access control classes are not used.
3. You add users plus resources of your own. The resources are protected by universal access rights as well as by individual access rights via classes.

In each of these three cases, a new DTSECTAB phase must be cataloged. z/VSE activates the new table when the next user identification and authentication is to be done. The source version of DTSECTAB is called DTSECTRC.

1. Predefined Security Support Only

You confine yourself to the resources defined in the pregenerated DTSECTAB. These resources are defined with UACC only; access control classes are not used. You just add/delete users. To do this, use one of the two methods:

- Define a user in the VSE.CONTROL.FILE, with the *Maintain User Profiles* dialog.
- Modify skeleton IESUPDCF (a member in library 59) and submit the job to batch. This job updates the VSE.CONTROL.FILE.

2. Adding Resources - Using the UACC Parameter Only

You add resources in addition of those defined in the pregenerated DTSECTAB. The resources are protected by universal access rights only. This implies that you need not modify the user profiles in the VSE.CONTROL.FILE.

Proceed as follows:

- Get yourself a copy of DTSECTRC from VSE/ICCF library 59 into your private VSE/ICCF library. **Whenever you change DTSECTAB, use the member you copied into your private VSE/ICCF library.**
- Update member DTSECTRC in your private VSE/ICCF library by adding definitions for the resources you want to protect.
- Submit DTSECTRC from your private VSE/ICCF library.

3. Adding Resources - With ACC Parameter

You add **resources** in addition to those defined in the pregenerated DTSECTAB. Some of the resources are protected via **access control classes**. This forces you to modify user profiles in the VSE.CONTROL.FILE because the ACC parameter must be included.

Proceed as follows for users:

- Use the *Maintain User Profiles* dialog as described under “Add or Update a User ID” on page 100.

For resources, proceed in the same way as described under “2. Adding Resources - Using the UACC Parameter Only.”

DTSECTRC Affected by IBM Service

Since DTSECTRC is code supplied by IBM, it can be affected by IBM service.

After a service PTF or FSU (Fast Service Upgrade) has been applied, you will have the latest IBM-supplied version of DTSECTRC in VSE/ICCF library 59. Consider that you may have to **update your version in sublibrary PRD2.SAVE** accordingly.

Migrating Old User and Resource Definitions in DTSECTAB

To migrate DTSECTAB definitions from a previous VSE release, proceed as follows:

1. Use the *Maintain User Profiles* dialog or batch utility IESBLDUP to redefine your current users. The dialog stores this information in the VSE.CONTROL.FILE.
2. Use a copy of DTSECTRC.A from PRD2.SAVE and add your old (and still valid) resource definitions to it. Remove the users still included in DTSECTRC.A since they are now included in the VSE.CONTROL.FILE (Step 1.)

Protecting the Access Control Table DTSECTAB

To prevent manipulation and misuse of the information stored in DTSECTAB, the IBM-supplied DTSECTAB has entries which serve to protect the DTSECTAB itself. Please refer to Figure 53 on page 171.

The system encrypts sensitive information in the table to provide additional protection.

VSE/ICCF Considerations

VSE/ICCF does not use the z/VSE security support to control the logon process and access to VSE/ICCF members.

In VSE/ICCF, access to VSE/ICCF libraries is controlled by defining a library as **public** or as **private**. In addition to a VSE/ICCF user's primary library, up to 8 alternate (private) libraries can be allocated to the user.

If SEC=NO ....

then VSE/ICCF uses tables of its own to control the access to batch resources such as files and programs in VSE libraries. The tables provided are:

- System Program Table
- Load Protection Table
- System File Table

Refer to the IBM manual *VSE/ICCF Administration and Operation* for details about VSE/ICCF libraries and protection tables under "Types of User Libraries" and under "Access Control Facilities".

If SEC=YES ....

then VSE/ICCF's protection mechanism is bypassed. z/VSE Access Control checks all accesses from VSE/ICCF interactive partitions in the same way as accesses from batch partitions via DTSECTAB.

Access checking for jobs running in interactive partitions uses the ID of the VSE/ICCF terminal user. Therefore, the VSE/ICCF terminal user must have a user profile entry in VSE.CONTROL.FILE. The passwords need not match because the user's authentication was already done at logon.

LIBDEF definitions in the CICS/ICCF partition startup are valid for interactive partitions and the terminal user. Therefore, it is recommended to keep the **permanent LIBDEFs** that are supplied by the pregenerated DTSECTAB (see also section "Access Control for LIBDEF Statements" on page 157). In this way, only the universal access rights are granted to interactive partition users.

Dummy Resource IJSYSRS.SYSLIB.DTSUTILA

This is a dummy resource which has an entry in DTSECTAB but is not a real member of any library. Users having READ access to this member are allowed to issue security-relevant DTSUTIL commands, for example ALTER USER. Other users may only read or write to VSE/ICCF members.

Mismatch of Passwords Between VSE/ICCF and Interactive Interface

Normally, a user's password under VSE/ICCF and the password under the VSE Interactive Interface are identical.

However, the passwords may differ, for example if the Interactive Interface user changed the password while VSE/ICCF was not active. Or, the terminal user changed the VSE/ICCF password via the /PASSWRD command.

With respect to VSE Access Control checking, this mismatch does no harm. In fact, if the two passwords differ, VSE/ICCF ″adapts″ the VSE/ICCF password to the Interactive Interface password when it detects the mismatch.

Logging and Reporting Accesses to DTSECTAB Resources

The z/VSE optional program *VSE/Access Control-Logging and Reporting* is needed if an installation plans to keep track of the usage of certain protected resources defined in DTSECTAB. The program supports the logging of attempts to use protected resources without authorization and, optionally, of any **authorized** access to protected resources. It provides formatted reports of the information logged.

For a detailed description of this program, refer to the IBM manual *VSE/Access Control-Logging and Reporting: Program Reference and Operations Guide*.

The access control support can be used without the logging and reporting program being installed.

Planning for Logging and Reporting

For each resource that is defined with access control class(es) in the access control table DTSECTAB, a logging option can be specified. Depending on the specification of that option, either **all accesses** to this resource, or only **access violations**, are logged. For further details refer to the description of the LOG parameter of the DTSECTAB macro on page 164.

The VSE/Access Control-Logging and Reporting program (ACLR) enables the security administrator to audit the access to system resources. The program logs access control related events and allows you to print them later for analysis.

Whenever a logging situation occurs the logging and reporting program writes a record on the log data set, a file on disk, to record the event. (When assembling and cataloging B-transients, it can happen that **two** records are written to the logging file. Take this into account when reading reports of such accesses.) The data sets used for logging need not be on dedicated disk volumes, but should be on separate volumes.

Note: You should protect the two log data sets (their file names are IJSYSL1 and IJSYSL2) in the access control table DTSECTAB. Define them without the ACC parameter. This ensures that only the ACLR program and a system administrator (SYSA) have access to those files.

Logging Access Attempts to Libraries

To produce accurate statistics on access to library entities (libraries, sublibraries and members), the following must be taken into account:

When a member is to be accessed, the access control checking is done at each level in the hierarchy (see Figure 54 on page 181). First of all, the access rights to library/sublibrary are valid that were established when the sublibrary was accessed (for example due to processing of a LIBDEF). When the user is attempting read access to the member and has the READ access right to the library in which the member is stored, access control does no checking at sublibrary or member level (see "Access Control for Libraries" on page 155).

The attempted access is not logged at sublibrary level, because the user was not attempting to read the sublibrary as a whole. It is not logged at member level either, because the Access Control Function stopped when it found the needed access right at library level. So this access is **not** logged, no matter what was specified in the LOG parameter for the member and sublibrary.

If, however, the user in this example had only the connect right to the library and sublibrary, access control checking would continue down to member level. The access would be logged (if the relevant access class is specified in the LOG parameter for the member; access violations are logged in any case).

In short, if all accesses are to be logged, the entity to be accessed must require a higher access right than those above it in the hierarchy, and the access rights to the entities above it must be insufficient to allow the attempted access. (It makes no difference whether the access rights are granted as individual rights in the user profile entry or as universal access rights in the resource profile entry of the access control table.)

Here are a few examples:

- For **all** accesses to a member to be logged, the access right to the library and sublibrary in which it is stored must be limited to **connect**. There will be an access check for each member access.

  As an example, consider the following generic member entry:

  ```
  IJSYSRS.SYSLIB.*,ACC=(1),LOG=(1)
  ```

  The user with ACC=(1,READ) can read all members, and each read is logged.
- For **update and alter** accesses to a member to be logged, the access right to the library and sublibrary must be **read** or lower. In this case, read accesses (to members) do not cause any access checking and hence no logging.

  For update/alter of a member, access checking will be done.

Due to the above checking mechanisms, an installation has to make a choice between satisfactory performance and detail of logging.

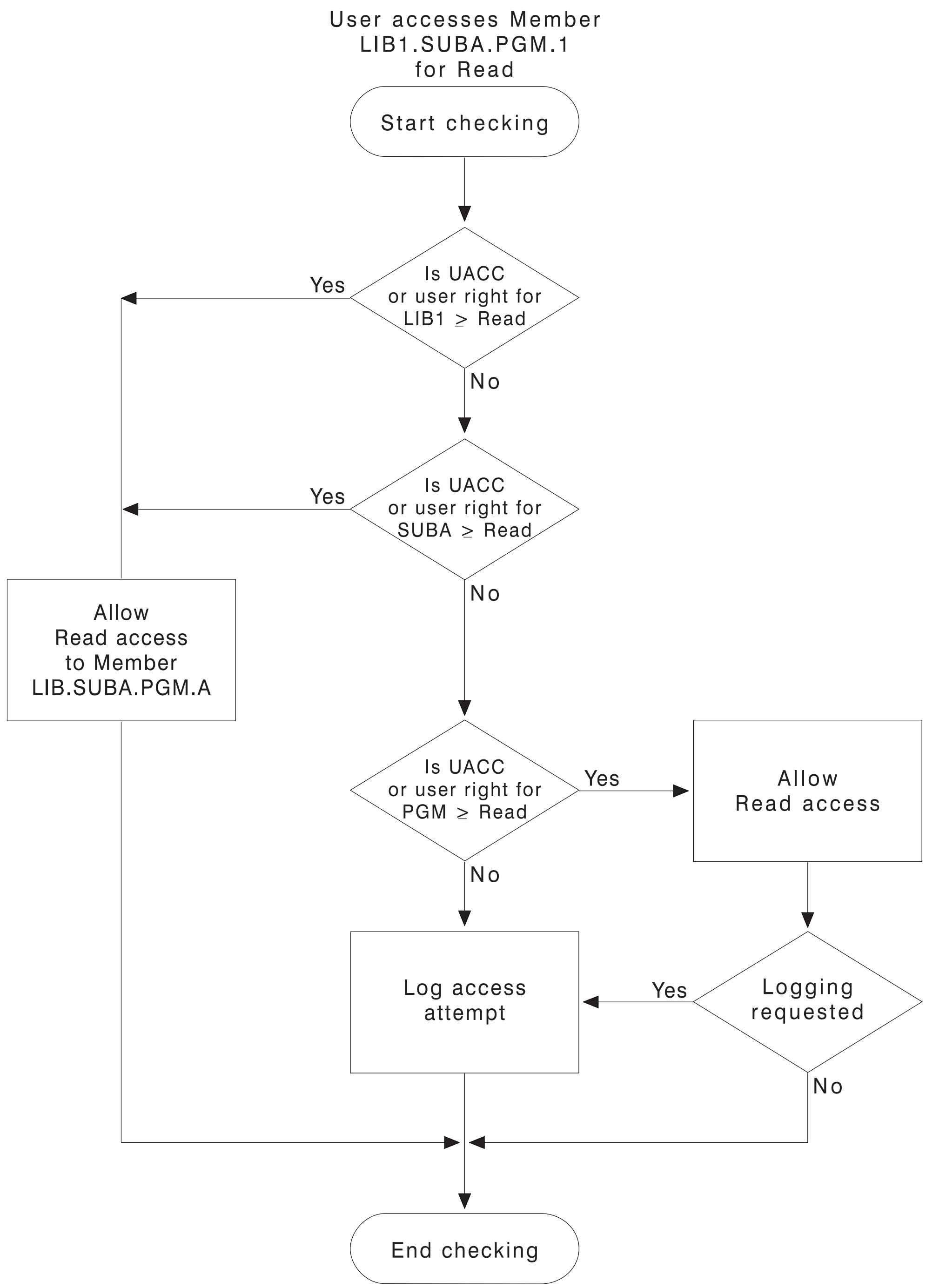
The same principle holds for the logging of accesses to sublibraries.

The security administrator should consider at which level each user should have access to the libraries, and grant only the **connect** access right to the level(s) above it. This will ensure accurate logging of access events.

The Reporting Module

The reporting module of the logging and reporting program creates printouts from the log data set according to your specifications. For more details on this topic, refer to "Auditing the Access to Controlled Resources" on page 182.

The reporting module must run in a batch partition.



Note that all accesses to libraries and files performed by the system administrator (SYSA) are logged, regardless whether the LOG parameter in the resource access definition was specified or not.

Figure 54. Access Logging when Accessing a Library Member

Auditing the Access to Controlled Resources

A record of unauthorized access attempts and attempts to access sensitive data can help the system administrator to identify data security problems. Auditing means analyzing these access attempts. The audit trail is the record of critical access attempts.

Careful analysis of the audit trail helps you to:

- Identify access violations and the user responsible for them;
- Find security weak points;
- Adapt access control measures to changing conditions;
- Recognize a need for corrective action by management;
- Use the system more efficiently.

VSE/Access Control-Logging and Reporting (ACLR), a z/VSE optional program, offers all of the above functions.

Hints for Auditing

Most access violations are the result of errors, and occur with a fairly constant frequency. Each system develops a characteristic pattern after a certain period of routine running. Any drastic change in this pattern in an otherwise unchanged system should be investigated.

For example, an unusually low number of violations may mean that someone has found a way to circumvent access control routines. The following examples give possible explanations for other symptoms:

- More people are authorized to work with the system and protected resources than are actually doing so: This may indicate that authorization is too easy to get without apparent necessity, or an anticipated necessity did not materialize.
- Many more authorized people work with the system and the protected resources than was anticipated: Again this may indicate that authorization is too easy to get, or that protection was defined for resources which actually need not be protected.
- Fewer people are authorized to use the system and protected resources than anticipated: This may indicate that authorization is too difficult to get, or the need to work with the system is not as high as estimated, or not all of the data that requires protected status has been defined.
- Some resources have an unexpectedly low access activity: It may indicate that the resource can possibly be removed from the system, or that an authorization is too difficult to get, or that a way has been found to circumvent protection.
- Some resources have an unexpectedly high access activity: In case of a file, it may indicate that a separation of sensitive and nonsensitive data should be considered to increase processing efficiency; for other resources, it may indicate that authorization is too easy to get.

These examples show that security measures must be reviewed frequently and be adjusted as the need arises.

The audit trail should be made available to the people and departments involved, and a list of violations should be distributed as often as possible to allow for quick reactions. A file owner, for example, should be notified if an unauthorized access to his file has been recorded.

How to Get an Audit Trail

The ACLR program enables the user, normally the security administrator, to get a printed audit trail from the information stored in the log data set. The reporting program (DSPRPM) produces printouts of the log data set, according to the selection criteria defined by the user in the ACCESS control statement of ACLR.

Unsuccessful identification and authentication attempts are not subject to logging by ACLR.

Operating a System with Security Active

This section presents a collection of security items to be considered in a system with security active.

Some General Rules

- The console must be physically protected as a **physical object** because anybody with physical access to the console can IPL the system with SEC=NO.
- Do not place jobs that access secured resources into the RDR queue while security is not yet active. Such jobs could fail after you IPL your system with security active. This is because the propagation of security information (user ID!) does not happen when security is not active.

  To avoid this potential problem, place explicit security information into the job.
- When using VSE/ICCF via the Interactive Interface, passwords may differ between the z/VSE Interactive Interface and VSE/ICCF without causing any harm. However, when invoking the VSE/ICCF utility DTSBATCH from the console, you must know the VSE/ICCF password.

Avoiding Startup Problems

You can start up your system even after a problem came up. Interrupt IPL processing with STOP=SVA and specify SEC=NO in the IPL SYS command. This lets your system work without security active.

Refer to the IBM manual *z/VSE Guide to System Functions* under “Interrupt and Restart the IPL Process” for information on how to interrupt IPL processing.

For recovery reasons it might be necessary to initialize your system without security support. In this case specify SEC=RECOVER in the IPL SYS command.

Performance Considerations

The impact on performance when using the z/VSE security support depends on how many resources are protected, how frequently they are accessed and by how many users, and whether all accesses are to be logged, or only the attempted violations.

Performance can be improved by efficient use of universal access rights. For example, a sublibrary whose members can be read (called for program execution) by anybody suggests itself for being protected only by universal access rights. In this case, access checking ends at sublibrary level; no time is wasted by access checking for each and every member.

Tape Handling

Tapes can be excluded from resource protection by specifying NOTAPE in the IPL SYS parameter SEC:

```
SEC=(YES,NOTAPE)
```

This is appropriate when, for example, you have a ″Tape Manager″ program installed that provides security checking for tapes.

In a system where tapes are part of the resource protection scheme, only standard-labeled tapes can be protected, and only under the following restriction. Specification of REWIND=NORWD in the associated DTFMT is allowed only if the tape is positioned at load point. Therefore, the tapes must be single-file. Otherwise the system could not verify whether the volume is on the tape.

In a system with security active, the following operational rules should be observed:

- The REWIND option must be specified for the particular magnetic tape volumes in the following VSE/VSAM commands: EXPORT, EXPORTRA, IMPORT, IMPORTRA, PRINT and REPRO. This means that multifile volumes on unlabeled tapes are not supported.
- The operator is not allowed to enter IGNORE as a reply to certain warning messages that relate to tape processing (these messages have numbers of type 41xx).

Controlling the Security Server Partition

The Security Server uses the VSE.CONTROL.FILE as data repository for user profiles. The Security Server runs per default in the FB partition.

For controlling the Security Server partition, specific MSG commands are available. For details on these commands refer to the manual *z/VSE Operation*.

If you have installed an External Security Manager (ESM), refer to the documentation shipped with this product.

Protecting z/VSE Resources Using Client Certificates and User IDs

If client authentication is required (in addition to server authentication), a *client certificate* is provided by clients to authenticate the client to the *server* (which in this case is the *z/VSE host*).

Using the service functions for client authentication, you can introduce access checking on client certificates via z/VSE User IDs that have been assigned to these client certificates. The client certificates belong to either CICS clients or VSE Connector Clients. Therefore, using client certificates you can control the access rights from:
- CICS clients to z/VSE host resources.
- VSE Connector Clients to z/VSE host resources.

A *client-certificate/User-ID mapping list* can be built and maintained using either the batch function BSSDCERT, or using the *Client-Certificates/User-IDs* dialog.

Refer to the *z/VSE e-business Connectors User's Guide* for:
- Explanations of the concepts of server authentication, client authentication, server certificates, client certificates, and root certificates.
- Details of how to set up the z/VSE host for Secure Sockets Layer (SSL) security.
- A detailed description of the *Client-Certificates/User-IDs* dialog.

Protecting z/VSE Resources Using Hardware Encryption

Hardware Encryption Assist Support (simply referred to as *Hardware Crypto support*) in z/VSE makes use of the following two hardware functions:
- Support of PCI Cryptographic Accelerator (PCICA) and Crypto Express2 (CEX2) cards. PCICA cards are available for all IBM zSeries processors, Crypto Express2 cards are available for zSeries processors z890, z990 and higher.
- Support of the CP Assist for Cryptographic Function (CPACF), which is part of the hardware of a z990 or higher.

PCICA and Crypto Express2 cards provide encryption-assist support, and can help to increase the throughput in a TCP/IP network using SSL (Secure Sockets Layer). CPACF provides hardware support for symmetric cryptographic algorithms, like DES, Triple-DES, and SHA-1.

Secure Sockets Layer (SSL) has become the dominant technique for enterprises to communicate securely with their customers via Internet browsers. SSL uses cryptography both for authentication of clients and servers, and for data confidentiality. SSL is a public key cryptography-based extension to TCP/IP networking. z/VSE supports the IBM Crypto Express2 and PCI Cryptographic Accelerator (PCICA) cards which provide encryption assist support for increased Secure Sockets Layer (SSL) throughput. The support is based on functions provided by *TCP/IP for VSE/ESA 1.5*.

The SSL support of *TCP/IP for VSE/ESA* was first introduced with VSE/ESA 2.6 and supported software encryption only. From VSE/ESA 2.7 onwards, SSL transparently uses PCICA cards if available. There is no need to change any applications already using SSL. For example, existing applications that use the SSL such as the CICS Web Support (CWS) and the VSE e-business connectors automatically benefit from this transparent use of PCICA cards.

For further details of Hardware Crypto support, refer to the chapter "z/VSE Security Support" in the manual *z/VSE Planning*.

Additional z/VSE Data Protection Facilities

z/VSE has several **standard facilities** for protecting data. These include:

- User-defined checking after IPL ("IPL Exit");
- User-defined checking of Job Control statements ("Job Control Exit");
- Identification and dating of files ("Labeling" on page 187);
- Protection of files on disk ("Data Secured Files" on page 187 and "Disk File Protection" on page 187);
- Prevention of concurrent update on disk files ("Track Hold Option" on page 188);
- The locking of resources using Assembler macros ("Data Integrity through Macros" on page 188).

IPL Exit

It may be desirable to perform certain integrity and security checks at the end of an IPL procedure: whether the right system pack was mounted, or whether the correct data was entered for the new work session. For example, if working with labeled files, it is important that the system date is correct, so as to provide for the protection of such files until their expiration date is actually reached.

After the IPL procedure has been completed, control is passed to the phase $SYSOPEN. This phase, a dummy phase in the IBM-supplied system sublibrary, can be replaced by a user exit routine to perform various checks that may be important for the security and integrity of the system. For more information on the IPL exit refer to the IBM manual *z/VSE Guide to System Functions* under "Writing an IPL Exit Routine".

Job Control Exit

This standard facility can be used for access control, but it requires a great deal of programming effort. It is better to use the access control table DTSECTAB for overall data protection in the system. You may, however, want to use the exit in special cases.

The job control exit transfers control to the phase $JOBEXIT each time a job control statement has been read. $JOBEXIT is a dummy phase in the IBM-supplied system sublibrary. It is loaded into the SVA automatically at IPL. You can replace the dummy phase by a user exit routine to perform various access control checks.

For example, to check the usage authorization for a program named PRIVLGE, the exit routine could be designed to examine the // EXEC statement for a specific code in the comment field as shown below:

```
User specification:
   // EXEC PRIVLGE   (no code specified as a comment)

Replacement by the user exit routine:
   // EXEC ERROR1
```

where program ERROR1 might simply issue a message indicating that the job cannot be processed due to a missing usage code.

The routine might also be used to examine an installation-defined identification code in the JOB statement, and compare this code with an access code associated with a file to be accessed.

The exit routine is not allowed to perform any I/O operations, to issue any SVCs, or to cancel the job.

For more information on the job control exit refer to the IBM manual *z/VSE Guide to System Functions* under "Writing a Job Control Exit Routine". This section also covers the setup of **multiple job control exits**.

Labeling

Labeling helps to ensure that the correct data is mounted for processing, and assists in protecting data against
- inadvertent destruction, and
- unauthorized access and usage.

This protection is provided for files stored on magnetic tape, disk, or diskette, where each file is identified by one or more file labels. In addition, each volume of data is identified by one volume label.

Volume and file labels are mandatory for disks and diskettes, and optional for tapes. For security reasons, however, it is strongly recommended to label all tapes that can use standard labels. Note, however, that some IBM programs require tapes without standard labels.

The TLBL statement is used for specifying label information for a magnetic tape, the DLBL and EXTENT statements for specifying it for a disk device, or a diskette. The VSE system label processing routines check whether the correct volume is mounted, and whether the retention period or expiration date has been reached. This protects files from being overwritten and destroyed prematurely.

The contents of the label must be specified when a file is created. The same label information must be available when the file is processed, to enable the system to compare the actual label of the data being processed with the label information submitted by the user. If a mismatch is detected the job is terminated.

For more information on labeling refer to the IBM manual *z/VSE Guide to System Functions* under "Job Control for Label Information".

Data Secured Files

The DSF operand in the DLBL statement indicates that a data secured file is to be created or processed. If a data secured file is to be accessed, a warning message is issued on the console. The operator then has to decide whether this file can be accessed by the program causing the message, and enter a reply at the console to allow the access. All these warning messages would make up a record of file accesses. While this method may have provided sufficient protection of and privacy for an installation's data in the past, it may not meet the protection and privacy standards of the future. Using the Access Control Function of VSE is the better method.

Disk File Protection

The DASD File Protection facility prevents programs from writing data outside the limits of their disk files. This might happen if, for example, a randomizing algorithm produces an unexpected disk address which is outside the file limits. Other files on a disk volume are thus protected against unintended destruction. However, if two disk files have been opened in the same partition and use the same programmer logical unit, these two files are not protected against destroying each other's data.

Disk file protection is activated if the DASDFP=YES operand is used in the IPL SYS command. DASDFP=YES is set in the IBM provided setting.

Track Hold Option

This facility is available only for disk volumes not under control of VSE/VSAM. In a multiprogramming environment, it prevents two or more programs from concurrently updating the same record of a track on a CKD-type disk volume, or the same record within a block on an FBA-type disk volume. In an installation that makes effective use of this facility, all of an installation's programs should specify track hold in their DTFs. Track hold support is available if the TRKHLD parameter in the FOPT supervisor generation macro was specified for supervisor assembly.

Data Integrity through Macros

In a multitasking environment, a mechanism is needed to prevent a task from using the resources of another task in an uncontrolled way, so as to avoid the destruction and erroneous updating of data. The lock management protects user-defined and system resources against concurrent use by different tasks in different partitions on one or more processors. Two levels of sharing are available when using the IBM-supplied LOCK and UNLOCK macros:

- Exclusive usage of a resource.
- Shared usage of a resource.

The following resources may be protected:

Files, libraries, catalogs, disk volumes and control blocks.

The resources are defined by symbolic names. Any symbolic name may be used; however, a naming convention should be established for the installation, and should be adhered to by all programmers using the LOCK and UNLOCK macros. A file name, a volume-id (VOLID), or a disk file begin address are examples of symbolic names.

The DTL and GENDTL macros are available to define a resource for share control. The DTL macro builds a lock control block, which the operating system needs to control the sharing of the particular resource during assembly; the GENDTL macro builds this block dynamically during execution.

Once the lock control block is defined for a resource, the operating system can efficiently control exclusive or shared access to the resource in accordance with the DTL or GENDTL macro. The MODDTL macro allows a lock control block to be modified dynamically.

A successful lock request (via the LOCK macro) means that the resource is locked for the task or partition issuing the request. With the UNLOCK macro the program can either release the resource completely, or in conjunction with the MODDTL macro, weaken the lock control from "exclusive control" to a shared status.

For more information on the various macros refer to the manuals *z/VSE System Macros Reference* and *z/VSE System Macros User's Guide*.

Protecting VSE/VSAM Files via Passwords

VSE/VSAM allows you to define passwords for accessing VSE/VSAM objects like:

Clusters
Alternate Indexes
Components (Data and Index)
Paths
Catalogs

To gain access to a protected object, a program or the operator must provide the password defined for it. You define passwords with the Access Method Services DEFINE command. Passwords can be defined for different levels of access:

1. Read access
2. Update access
3. Control-interval access
4. Full access

Moreover, you can supply a user security-verification routine to double-check the authority of a program accessing a file.

Note: The job streams created by the Interactive Interface of z/VSE (for File and Catalog Management) do not include VSE/VSAM data protection parameters. If you want to define passwords, for example, you must edit the job streams and add the required password parameter yourself.

For a detailed description of VSE/VSAM data protection, refer to the manual *VSE/VSAM User's Guide and Application Programming* under "Data Protection and Integrity Options." For a description of the DEFINE command refer to the manual *VSE/VSAM Commands* under "Defining a Catalog." In a z/VSE user profile you can assign the default catalog a user is allowed to access. In addition, you can assign authority for managing VSE/VSAM files or catalogs or both. Refer to "Maintaining User Profiles" on page 98 for details about maintaining user profiles.

Chapter 8. Installing a Second Predefined CICS Transaction Server

z/VSE provides support for installing a second, predefined CICS Transaction Server. The following text uses also the short form CICS for CICS Transaction Server.

The following skeletons are provided:

SKCICS2 (for defining startup)

SKPREPC2 (for defining resources)

The second CICS Transaction Server under z/VSE does not include VSE/ICCF and can have any of four relationships to the primary CICS:

1. No communication with the primary CICS.
2. Communication to the primary CICS via *Multiregion Operation* (MRO). The two CICS Transaction Servers run in the same processor.
3. Each CICS Transaction Server can communicate via MRO with CICS/VSE 2.3.
4. The two CICS Transaction Servers can also communicate via *Intersystem Communication* (ISC), running on the same or different processor. With ISC any type of CICS subsystem can communicate with each other.

z/VSE's support addresses only the first two cases, and so does this chapter. For information on ISC, consult the IBM manual *CICS Intercommunication Guide*.

Before you begin with the installation process you should consult the manual *z/VSE Planning* under "Planning for the Second CICS Transaction Server" for planning information.

The *z/VSE Planning* manual also provides detailed information about the "CICS Transaction Server Monitoring and Statistics Support".

Installation Tasks for a Second CICS Transaction Server

The tasks to be performed are described below. Follow the given sequence.

Task 1: Modify Predefined Environment

The following is assumed:

- You selected predefined environment B or C for initial installation.
- Your second CICS is to run in partition F8.

  This means that you can use skeleton SKALLOCB or SKALLOCC to change the partition values as shown below.
  - For environment B, the allocation reserves the following space in F8 for the second CICS:
    ```
    ALLOC F8=50M
    SIZE F8=2M
    ```
  - For environment C, the allocation reserves the following space in F8 for the second CICS:
    ```
    ALLOC F8=512M
    SIZE F8=2M
    ```


If you decide not to use F8, you must select a partition of the appropriate size and modify the startup job accordingly.

Note that these are recommended average values. Depending on your applications they may not be sufficient.

For the second, predefined CICS the name of the startup job is CICS2, the name of the corresponding skeleton is SKCICS2.

Before you edit the skeletons, copy them first from VSE/ICCF library 59 to your primary library.

Increasing partition sizes may also mean that to meet your total system requirements you have to either:
- Increase the VSIZE.
- Select the predefined environments B or C.

The predefined page data set size (VSIZE) for:
- Predefined environment B is 264 MB.
- Predefined environment C is 2 GB.

However, you can change the VIO and VSIZE values using the *Tailor IPL Procedure* dialog. Refer to "Tailoring the IPL Procedure" on page 8 for details.

In addition to running a CICS Transaction Server in a static partition, you may also run it in a **dynamic partition** of sufficient size:
- Predefined environment B allows for 50 MB partitions.
- Predefined environment C allows for 512 MB partitions.

However, you should adjust the value of EDSALIM in the DFHSIT table or the startup override accordingly:
- Predefined environment B has a predefined EDSALIM size of 25 MB.
- Predefined environment C has a predefined EDSALIM size of 450 MB.

Note also that the partition parameters and values for dynamic partitions are defined in the active dynamic class table. You can modify this table with the *Maintain Dynamic Partitions* dialog. For details refer to "Defining Dynamic Class Tables" on page 71.

Running your second CICS in another static partition than F8 (or in a dynamic partition) requires changes in skeletons SKCICS2 and SKPREPC2. Affected are mainly statements which include partition-related information. Further details are provided with the skeleton descriptions.

Do not run the CICS Transaction Server in partition F4 because of possible storage key problems if storage protection is set in DFHSIT.

Note: Before you submit skeletons SKCICS2 and SKPREPC2 later, ensure that you did run at least once skeleton SKCOLD. It updates procedure COLDJOBS which loads jobs into the VSE/POWER reader queue that are important for a COLD startup.

Task 2: Modify Skeletons Provided by z/VSE

Copy a skeleton first from VSE/ICCF library 59 to your primary library and modify the skeleton here. This ensures that you have a backup version of the original skeleton available.

Skeleton SKUSERBG

1. Locate the statement

```
* // PWR PRELEASE RDR,CICS2
```

2. Delete the asterisk and the blank in the first two columns. Refer also to Figure 12 on page 42.

   When the modified procedure is processed during system startup, the statement causes the second CICS subsystem to be started.

Skeletons SKCICS2 and SKPREPC2

Refer to “Skeleton SKCICS2” on page 197 and to “Skeleton SKPREPC2” on page 200 where the skeletons are shown in detail. Comments point out what might or should be modified and why a modification should be considered.

Modify the skeletons **but do not submit** them now.

Task 3: Modify CICS Control Tables

The control tables that may need to be modified for the second CICS subsystem are the following (their source can be obtained from VSE/ICCF library 59):
System Initialization Table (DFHSITC2)
Destination Control Table (DFHDCTC2)
File Control Table (DFHFCTC2)

The FCT will be migrated to te CSD file and further modifications should be done there.

Differences of significance are indicated in subsequent paragraphs that discuss the various tables. These paragraphs point out possible modifications. Some changes have to be made to the tables to include or exclude CICS-to-CICS communication support via MRO. You may also add entries to meet local requirements.

If your modified skeleton SKPREPC2 includes DLBL and EXTENT statements for CICS **Journal Files**, you must provide the specifications for the corresponding Journal Control Tables (DFHJCTs). You can use skeleton DFHJCTSP, stored in VSE/ICCF library 59, for that purpose. The skeleton reflects the journal control tables for the primary CICS. Your modification consists of changing the suffix ’SP’ into ’C2’ wherever it occurs. For a description of the DFHJCT macro, refer to the CICS Transaction Server manual *CICS Resource Definition Guide*.

System Initialization Table

This table, shipped as member **DFHSITC2**, **includes** significant differences compared to DFHSITSP as follows:

– Except for PLTPI and PLTSD, the table suffix is C2. Note that FCT is NO. Affected operands are:

```
DCT=C2      PLTPI=P2
FCT=NO      PLTSD=S2
```

– The application name of the CICS subsystem:

```
APPLID=PRODCICS
```

  PRODCICS is also the user ID defined in the VSE.CONTROL.FILE and used in the ID statement as user ID for startup.

– The table activates the spool support of CICS with the specification

```
SPOOL=(YES,B,A)
```

– The internal trace function is set to off (the second CICS Transaction Server is assumed to support applications in production):

```
TRTABSZ=256
TRTRANSZ=128
TRTRANTY=TRAN
```

– As shipped, table DFHSITC2 does not activate CICS-to-CICS communication via MRO. The related specifications are set as follows:

```
GRPLIST=VSELST2
IRCSTRT=NO
ISC=YES
SYSIDNT=CIC2
```

SKPREPC2 defines list VSELST2.

If **MRO communication is to be used**, IRCSTRT=NO has to be changed into IRCSTRT=YES (in table DFHSITC2). In table DFHSITSP for the primary CICS system, the settings for MRO must also be changed from NO to YES:

```
IRCSTRT=YES
ISC=YES
```

Destination Control Table

This table is shipped as member **DFHDCTC2**; it includes no significant differences. Any TYPE=SDSCI entries that you need in addition are to be added immediately behind the box labeled

```
LOCAL ENTRIES FOR TYPE=SDSCI SHOULD BE PLACED BELOW THIS BOX
```

File Control Table

This table is shipped as member **DFHFCTC2**; it includes no significant differences. Any entries that you need in addition are to be added immediately behind the box labeled

```
LOCAL ENTRIES SHOULD BE PLACED BELOW THIS BOX
```

The default FCT (DFHFCTC2) is migrated into the CSD (CICS System Definition) file (as group FCTC2) and FCT=NO is set. Thus modifying DFHFCTC2 does not change the system unless the FCT is migrated to the CSD after changing it. Migration is done using utility DFHCSDUP:

```
// EXEC DFHCSDUP,SIZE=600K
   DELETE GROUP(FCTC2)
   MIGRATE TABLE(DFHFCTC2)
/*
```

The initial setup using SKPREPC2 will migrate the DFHFCTC2.

You may also use RDO (CEDA command) for defining FCT entries.

Task 4: Submit the Modified Skeletons

After having modified the skeletons as described under “Skeleton SKCICS2” on page 197 and “Skeleton SKPREPC2” on page 200, submit the skeletons from the FULIST of your primary VSE/ICCF library.

Ensure to submit them in the sequence shown:

1. SKCICS2 – use option **7**.

   Before you submit the next skeleton, close the Message Routing file. Use the CICS command:

```
CEMT SET FILE(IESROUT) CLOSE
```

If you define the Workstation File Transfer Support for your second CICS subsystem, close also the Host Transfer file. Use the CICS command:

```
CEMT SET FILE(INWFILE) CLOSE
```

2. SKPREPC2 – use option **7**.
3. The CICS control tables that you modified or coded – use option **7**.
4. When processing of these skeletons is complete, reopen the file(s) that you have closed. Use the CICS command(s):

```
CEMT SET FILE(IESROUT) OPEN ENA
CEMT SET FILE(INWFILE) OPEN ENA
```

There is no need for you to define any terminals to the second CICS. Also, the name of the second CICS (PRODCICS) is already defined to VTAM.

If **MRO communication is to be used**, however, you should assign unique CICS terminal IDs to the terminals of the second CICS. Refer also to “Tailoring Autoinstall Terminals” on page 196.

Task 5: Definitions for MRO

For this task use the RDO (Resource Definition Online) function described in the CICS Transaction Server manual *CICS Resource Definition Guide*. The function is a convenient means for setting up a communication path to the primary CICS subsystem and for defining terminals.

1. **Define CICS-to-CICS MRO communication**

   This requires the definition of a connection and an associated sessions definition for each of the two CICS subsystems.

   For a **connection** definition, enter the RDO command

   ```
   CEDA DEFINE CONNECTION
   ```

   and provide specifications as listed below. Accept the defaults for the data-entry fields not listed here.

Panel Line	Specifications DBDCCICS Side	Specifications PRODCICS Side	Comment
Connection:	CIC2	CIC1	SYSID in DFHSITC2
Group:	VSEIRC1	VSEIRC2	
Netname:	PRODCICS	DBDCCICS	
ACcessmethod:	IRc	IRc	
Protocol:			Must be blank
AUToconnect:	Yes	Yes	

   For the associated **sessions** definition, enter the RDO command

   ```
   CEDA DEFINE SESSIONS
   ```

   Provide the specifications listed below to have 10 send and receive sessions. Accept the defaults for the data-entry lines not listed here.

Panel Line	Specifications DBDCCICS Side	Specifications PRODCICS Side	Comment
Sessions:	CICSS2	CICSS1	arbitrary

Panel Line	Specifications DBDCCICS Side	Specifications PRODCICS Side	Comment
Group:	VSEIRC1	VSEIRC2	
Connection:	CIC2	CIC1	SYSID in DFHSITC2
Protocol:	LU61	LU61	
RECEIVEPfx:	TR	PR	
RECEIVECount:	010	010	
SENDPfx:	TS	PS	
SENDCount:	010	010	
SENDSize:	4096	4096	See Note a below
RECEIVESize:	4096	4096	See Note a below
OPERRsl:	0	0	See Note b below
OPERSecurity:	1	1	See Note b below
AUToconnect:	Yes	Yes	
INservice:	Yes	Yes	
RELreq:	Yes	Yes	
Discreq:	Yes	Yes	

Notes:

a. A general recommendation: certain CICS applications may require specific values to be specified. Check the applicable manuals.

b. This is the default: gives the terminal operator access to unprotected resources only.

2. **Define terminals**

   Use the autoinstall function of CICS to define the terminals that are to be supported by your second CICS.

   Enter RDO commands as follows:

   ```
   CEDA ADD GROUP(VSEIRC2) LIST(VSELST2)
   ```

   The statements cause all of list VSELIST to be copied into the new CSD list, and the new group VSEIRC2 (which you had defined via CEDA DEFINE SESSIONS), to be added to the list.

   ```
   CEDA ADD GROUP(VSEIRC1) LIST(VSELIST)
   CEDA INSTALL GROUP(VSEIRC1)
   ```

   The above statements add and install the new CSD group, `VSEIRC1`, which you had defined via CEDA DEFINE SESSIONS for your primary CICS.

After successful completion of the above procedure, the required definitions for your second CICS are complete. At the next startup of your z/VSE system, your second CICS will be available.

Tailoring Autoinstall Terminals

For using unique CICS terminal IDs, you have to perform the following steps:

- In the *Hardware Configuration* dialog select option 3 for logical unit (further processing) for autoinstall terminals.

- If the entry for CICS TERM ID is displayed, use option 6 and delete the entry for TERM ID (CICS).

The TERM IDs (such as A001, A002 and so on) are used for the first and a second CICS.

To prevent the use of duplicated CICS TERM IDs for the second CICS, do the following:

- Access and copy the autoinstall exit member IESZATDX from VSE/ICCF library 59 to your own VSE/ICCF library.
- Locate the field **`PREFIX DC 'A'`**.
- Change the letter 'A' to any other letter (for example, 'B') which is to be used as prefix for the CICS TERM ID.
- Submit the changed member IESZATDX.
- Ensure that the phase is being cataloged into a sublibrary unique to the second CICS Transaction Server.

Considerations for Problem Solving

As shipped, the startup job stream (skeleton SKCICS2) defines and allocates an AUXTRACE file. Yet, tracing the flow of transactions through the system is not activated automatically. For how to use AUXTRACE, refer to the CICS Transaction Server manual *CICS Problem Determination Guide*. Skeleton DFHAUXPR, which is stored in VSE/ICCF library 59, provides a job stream for analyzing AUXTRACE data. The label information in the job stream must be adapted for the second CICS (PRODCICS).

Skeletons for Second CICS Transaction Server

This appendix lists and describes skeletons:
SKCICS2
SKPREPC2

These skeletons are shipped as members of VSE/ICCF library 59. The skeletons as listed in this section include comments that you may find helpful when modifying them.

Also note that names of resources (such as volumes) that you might want or have to change are highlighted in bold.

Skeleton SKCICS2

This is the startup procedure for the second CICS Transaction Server. You can submit this skeleton unchanged if your second CICS Transaction Server is to run in partition F8. Else, change the highlighted specifications accordingly and ensure that XAPPLF8 in CPUVAR1 is modified accordingly.

The first loading of CICS2 is done via skeleton SKPREPC2. If there is a need to load CICS2 again, you have to remove the asterisk (*) in front of the command `EXEC PROC=LDCICS2` to activate the loading into the VSE/POWER reader queue.

After you have modified skeleton SKCICS2, enter the following command from the editor's command line **before** you file the skeleton:

`@DTRSEXIT`

This command calls a macro that deletes specific comments from the skeleton.

```
* $$ JOB JNM=CATCICS2,DISP=D,CLASS=0
// JOB CATCICS2                  CATALOG CICS2 AND LDCICS2
// EXEC LIBR,PARM='MSHP'
   ACCESS S=IJSYSRS.SYSLIB
   CATALOG CICS2.Z REPLACE=YES
$$$$ JOB JNM=CICS2,DISP=L,CLASS=8,EOJMSG=YES
$$$$ LST CLASS=A,DISP=D,RBS=100
// JOB CICS2         STARTUP OF SECOND CICS WITHOUT ICCF
// OPTION SADUMP=5
// OPTION SYSDUMPC
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASED,PRD1.BASE,PRD2.PROD,        X
               PRD2.SCEEBASD,PRD2.SCEEBASE,PRD2.DBASE),PERM
// LIBDEF DUMP,CATALOG=SYSDUMP.F8
// SETPARM XNCPU=''
// SETPARM XMODEF8=AUTO
// SETPARM XAPPLF8=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF8=ACTIVE' **F8 ASSUMED
$$/*
// EXEC PROC=CPUVAR&XNCPU,XMODEF8,XAPPLF8        **F8 ASSUMED
// SETPFIX LIMIT=256K
LOG
// ID USER=PRODCICS
NOLOG
// EXEC PROC=DTRCICS2                  LABELS FOR CICS FILES
*   WAITING FOR VTAM TO COME UP
// EXEC IESWAITT
$$/*
*   WAITING FOR TCP/IP TO COME UP
* // EXEC REXX=IESWAITR,PARM='TCPIP00'
$$/*
// ASSGN SYS020,SYSLST
* // ASSGN SYS023,DISK,VOL=SYSWK1,SHR   IF JOURNALLING
// IF XENVNR = B THEN
// SETPARM ELIM=25M
// IF XENVNR = C THEN
// SETPARM ELIM=450M
```

Comment

In a system with security (access control) active the // ID statement (for user PRODCICS) ensures that CICS2 has the appropriate access rights to the control file. When you submit the job your access rights are inherited by the CICS2 startup job in the VSE/POWER reader queue, provided that the // ID statement is for PRODCICS or is of the same model type like DBDCCICS or PRODCICS. In this case, no password is required. To get inheritence the job must be submitted when security is active.

Chapter 7, "Protecting Resources," on page 129 provides details about the z/VSE access control support.

If you use another name than CICS2, you must also update procedures such as USERBG and COLDJOBS (via skeletons SKUSERBG and SKCOLD).

ELIM is the value of EDSAMIM. For environment C, the specified value requires a partition of at least 480 MB.

Figure 55. Skeleton SKCICS2 (Starting Up Second CICS in Partition F8) (Part 1 of 2)

```
// IF XMODEF8 = COLD THEN
// GOTO COLDST
// SETPARM XMODEF8=AUTO
// GOTO STARTCIC
/. COLDST
// SETPARM XMODEF8=COLD
/. STARTCIC
// EXEC DFHSIP,SIZE=DFHSIP,PARM='APPLID=&XAPPLF8.,START=&XMODEF8.,SI', *
               DSPACE=2M,OS390
SIT=C2,STATRCD=OFF,NEWSIT=YES,
$$$$ SLI MEM=IESVAEXC.Z,S=IJSYSRS.SYSLIB
$$/*
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF8=INACTIVE'
$$/*
$$/&
$$$$ EOJ
/+
CATALOG  LDCICS2.PROC     REPLACE=YES DATA=YES
// EXEC DTRIINIT
   LOAD CICS2.Z
/*
/+
/*
* // EXEC PROC=LDCICS2       LOAD CICS2 INTO RDR QUEUE
/*
/&
* $$ EOJ
```

Figure 55. Skeleton SKCICS2 (Starting Up Second CICS in Partition F8) (Part 2 of 2)

The skeleton includes additional $$ characters. They are needed to mask off VSE/POWER JECL statements. Program DTRIINIT, described in the manual *z/VSE System Utilities* under “DTRIINIT Utility”, replaces the $$ characters with VSE/POWER JECL statements for cataloging.

Skeleton SKPREPC2

Note: Before you submit SKPREPC2 for processing, you may have to change skeleton SKUSERBG in VSE/ICCF library 59. See also the "Comment" on page 205.

The skeleton defines the resources for a second CICS Transaction Server and catalogs the required label information.

For space requirements, consult the manual *z/VSE Planning*.

```
* $$ JOB JNM=VSAMDEF2,DISP=D,CLASS=0
// JOB VSAMDEF2 - DEFINE VSAM CLUSTERS FOR SECOND CICS
* ******************************************
*  DEFINE AND INITIALIZE VSAM FILES FOR CICS2
* ******************************************
// EXEC IDCAMS,SIZE=AUTO
 /*                                                             */
 /*  DELETE VSAM FILES                                          */
 /*                                                             */
```

Comment

The delete job below ensures that no catalog entries with identical file IDs exist.

You might have files with an ID identical to the ones specified in the job and also under control of the specified user catalog. If you need those files further on (to operate with three CICS Transaction Servers, for example), rename the IDs in the skeleton. Suggested approach: change `CICS2` to `CICSB`.

If you do not use the default user catalog, do a global change for the catalog name, **VSESPUC**, and the catalog ID, **VSESP.USER.CATALOG**. The occurrences of these specifications in the skeleton are highlighted.

```
   DELETE (CICS2.GCD) CL NOERASE PURGE -
   CATALOG(VSESP.USER.CATALOG)
   DELETE (CICS2.LCD) CL NOERASE PURGE -
   CATALOG(VSESP.USER.CATALOG)
   DELETE (CICS2.ONLINE.PROB.DET.FILE) CL NOERASE PURGE  -
   CATALOG(VSESP.USER.CATALOG)
   DELETE (CICS2.DFHTEMP) CL NOERASE PURGE  -
   CATALOG(VSESP.USER.CATALOG)
   DELETE (CICS2.TD.INTRA) CL NOERASE PURGE  -
   CATALOG(VSESP.USER.CATALOG)
   DELETE (CICS2.RSD) CL NOERASE PURGE  -
   CATALOG(VSESP.USER.CATALOG)
   SET MAXCC = 0
 /*                                                             */
 /*  DEFINE VSAM FILES                                          */
 /*                                                             */
```

Figure 56. Skeleton SKPREPC2 (Part 1 of 7)

Comment

The required VSE/VSAM files are defined to reside on the SYSWK1 volume with DOSRES specified as the secondary volume for allocations. If you plan to have the files allocated in VSE/VSAM space on different volumes, change the volume names accordingly.

```
DEFINE CLUSTER(NAME(CICS2.GCD)           -
    RECORDSIZE (4089 4089)               -
    RECORDS (2000 200)                   -
    KEYS   (28 0)                        -
    REUSE                                -
    INDEXED                              -
    FREESPACE (10 10)                    -
    SHR(2)                               -
    CISZ(8192)                           -
    VOL(SYSWK1 DOSRES))                         -
    DATA(NAME(CICS2.GCD.@D@))            -
    INDEX (NAME (CICS2.GCD.@I@))         -
    CATALOG(VSESP.USER.CATALOG)
 DEFINE CLUSTER(NAME(CICS2.LCD)          -
    INDEXED                              -
    RECORDSIZE (45 124)                  -
    RECORDS (3000 200)                   -
    KEYS   (28 0)                        -
    REUSE                                -
    FREESPACE (10 10)                    -
    SHR(2)                               -
    CISZ(2048)                           -
    VOL(SYSWK1 DOSRES))                         -
    DATA(NAME(CICS2.LCD.@D@))            -
    INDEX (NAME (CICS2.LCD.@I@))         -
    CATALOG(VSESP.USER.CATALOG)
 DEF CLUSTER(NAME(CICS2.ONLINE.PROB.DET.FILE) -
     FILE(IESPRB)                         -
     VOL(SYSWK1 DOSRES)                   -
     RECORDS (300 100)                    -
     RECORDSIZE (4000 4089)               -
     INDEXED                              -
     KEYS(2 0)                            -
     SHR(2))                              -
     DATA (NAME (CICS2.ONLINE.PROB.DET.FILE.@D@) CISZ(4096)) -
     INDEX (NAME (CICS2.ONLINE.PROB.DET.FILE.@I@) CISZ(512)) -
     CATALOG(VSESP.USER.CATALOG)
 /*                                                            */
```

Figure 56. Skeleton SKPREPC2 (Part 2 of 7)

```
DEF CLUSTER(NAME(CICS2.DFHTEMP)            -
     VOL(SYSWK1 DOSRES)                     -
     RECORDS (100)                          -
     RECORDSIZE (16377 16377)               -
     CISZ   (16384)                         -
     NONINDEXED                             -
     SHR(2))                                -
     DATA(NAME(CICS2.DFHTEMP.ESDS))         -
     CATALOG(VSESP.USER.CATALOG)
 /*                                                              */
 DEF CLUSTER(NAME(CICS2.TD.INTRA)           -
     VOL(SYSWK1 DOSRES)                     -
     RECORDS (100)                          -
     RECORDSIZE (4089 4089)                 -
     CISZ   (4096)                          -
     NONINDEXED                             -
     SHR(2))                                -
     DATA(NAME(CICS2.TD.INTRA.ESDS))        -
     CATALOG(VSESP.USER.CATALOG)
 /*                                                              */
 DEF CLUSTER(NAME(CICS2.RSD)                -
     INDEXED                                -
     RECORDSIZE (2000 2000)                 -
     RECORDS (250 100)                      -
     KEYS   (22 0)                          -
     FREESPACE (20 20)                      -
     SHR(2)                                 -
     VOL(SYSWK1 DOSRES))                    -
     DATA(NAME(CICS2.RSD.@D@))              -
     INDEX (NAME (CICS2.RSD.@I@))           -
     CATALOG(VSESP.USER.CATALOG)
 /*                                                               */
```

Comment

The following files are shared with CICSICCF, the primary CICS Transaction Server and need not be defined:

VSE.TEXT.REPOSITORY.FILE
VSE.MESSAGE.ROUTING.FILE
CICS.CSD
VSE.CONTROL.FILE

```
/*
// IF $RC > 0 THEN
CANCEL
```

Figure 56. Skeleton SKPREPC2 (Part 3 of 7)

```
*
*    INITIALIZE THE CICS2 RESTART DATA SET
*
// DLBL DFHRSD,'CICS2.RSD',0,VSAM,CAT=VSESPUC
// DLBL DFHGCD,'CICS2.GCD',0,VSAM,                                      X
               CAT=VSESPUC
// EXEC IDCAMS,SIZE=AUTO
   REPRO INFILE   -
        (SYSIPT                 -
         ENVIRONMENT            -
          (RECORDFORMAT (FIXUNB) -
           BLOCKSIZE(80)        -
           RECORDSIZE (80)))    -
        OUTFILE (DFHRSD)
ACTL 0001
/*
// EXEC IDCAMS,SIZE=AUTO          INIT GCD FILE
   REPRO INFILE                                        -
           (SYSIPT                                     -
           ENVIRONMENT                                 -
             (RECORDFORMAT(FIXUNB)                     -
              BLOCKSIZE(80)                            -
              RECORDSIZE(80)))                         -
         OUTFILE(DFHGCD)
/*
// DLBL DFHLCD,'CICS2.LCD',0,VSAM,                                      X
                CAT=VSESPUC
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD2.SCEEBASE,PRD1.BASE)
// EXEC DFHCCUTL,SIZE=300K        INITIALIZE CICS CATALOG
/*
/&
* $$ EOJ
* $$ JOB JNM=DTRCICS2,DISP=D,CLASS=0
// JOB DTRCICS2 - DEFINE LABELS FOR SECOND CICS
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG DTRCICS2.PROC  D=YES  R=YES  EOD=/+
// ASSGN SYS018,DISK,VOL=SYSWK1,SHR
// DLBL DFHDMPA,'CICS2.DUMPA',0,VSAM,
               CAT=VSESPUC,RECSIZE=7200,
               DISP=(NEW,KEEP),RECORDS=(300,0)
// DLBL DFHDMPB,'CICS2.DUMPB',0,VSAM,
               CAT=VSESPUC,RECSIZE=7200,
               DISP=(NEW,KEEP),RECORDS=(100,0)
// DLBL DFHAUXT,'CICS2.AUXTRACE',0,VSAM,
               CAT=VSESPUC,RECSIZE=4096,
               DISP=(NEW,KEEP),RECORDS=(400,0)
// DLBL DFHTEMP,'CICS2.DFHTEMP',0,VSAM,
               CAT=VSESPUC
// DLBL DFHNTRA,'CICS2.TD.INTRA',0,VSAM,
               CAT=VSESPUC
// DLBL DFHRSD,'CICS2.RSD',0,VSAM,
               CAT=VSESPUC
```

Figure 56. Skeleton SKPREPC2 (Part 4 of 7)

```
// DLBL DFHLCD,'CICS2.LCD',0,VSAM,
               CAT=VSESPUC
// DLBL DFHGCD,'CICS2.GCD',0,VSAM,
               CAT=VSESPUC
// DLBL IESPRB,'CICS2.ONLINE.PROB.DET.FILE',,VSAM,
               CAT=VSESPUC
* ******************************************************************* C
* REMOVE COMMENTS AND TRAILING 'C' IN COLUMN 71 IN CASE JOURNALLING   C
* IS USED. ADJUST LABELS AS SPECIFIED IN SKJOUR2 - DEPENDING ON YOUR  C
* DISK TYPE.                                                          C
*                                                                     C
*  DLBL DFHJ01A,'CICS2.SYSTEM.LOG.A',0,SD                             C
*  EXTENT SYS023,DOSRES,1,0,XXXX,60                                   C
*  DLBL DFHJ01B,'CICS2.SYSTEM.LOG.B',0,SD                             C
*  EXTENT SYS023,DOSRES,1,0,XXXX,60                                   C
*  DLBL DFHJ02A,'CICS2.USER.JOURNAL.A',0,SD                           C
*  EXTENT SYS023,DOSRES,1,0,XXXX,60                                   C
*  DLBL DFHJ02B,'CICS2.USER.JOURNAL.B',0,SD                           C
*  EXTENT SYS023,DOSRES,1,0,XXXX,60                                   C
* ******************************************************************* C
/+
  CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
  COPY DTRCICS2.PROC  REPLACE=YES
/*
/&
* $$ EOJ
```

Comment

If you plan to define and format one or more CICS journal files, insert the required DLBL and EXTENT statements at this point. Member SKJOURN of the VSE/ICCF library 59 includes suitable sample statements for all types of the supported IBM disk devices. For further details about defining journal files, refer to "Task 3: Modify CICS Control Tables" on page 193.

The SETPARM values specified in the job below assume that your second CICS will run in partition F8. If this is your intent, no partition-related specifications need be changed. However, if your second CICS Transaction Server is to run in another partition, F5 for example, change the SETPARM values as shown:

Current Specification	Change To
XPARTC2='F8'	XPARTC2='F5'
XUSEF8='CI'	XUSEF5='CI'
XAPPLF8='PRODCICS'	XAPPLF5='PRODCICS'

Do not use the partition F4 because of possible storage key problems if storage protection is set in DFHSIT. Also, be aware of the size requirements of the partition.

Modify the job if necessary. If journaling is used, the journal files are located on DOSRES. However, for environment C you should *locate the journal files to a disk other than DOSRES*. Otherwise, the page data set might be overwritten!

```
* $$ JOB JNM=GLOBVAR,DISP=D,CLASS=0
// JOB GLOBVAR - DEFINE GLOBAL VARIABLES
// SETPARM XNCPU='
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU'
   SET XPARTC2='F8'
   SET XUSEF8='CI'
   SET XAPPLF8='PRODCICS'
/*
/&
* $$ EOJ
```

Figure 56. Skeleton SKPREPC2 (Part 5 of 7)

Comment

The following job replaces the currently used startup procedure USERBG for the BG partition. In the SLI statement, replace the library number 59 by the number of your primary library (assuming that you had applied the changes to SKUSERBG in your primary library). Jobs PRTDUC2A and PRTDUC2B are for printing the dump data sets of CICS.

If your currently used startup procedure includes changes of your own, ensure that you include your own member in the SLI statement instead of member SKUSERBG.

```
* $$ JOB JNM=COPYUBG,DISP=D,CLASS=0
// JOB COPYUBG - COPY SKUSERBG FROM ICCF LIBRARY
// EXEC LIBR,PARM='MSHP'
   ACCESS S=IJSYSRS.SYSLIB
* $$ SLI ICCF=(SKUSERBG),LIB=(59)
/*
/&
* $$ EOJ
* $$ JOB JNM=ADDJOBS,CLASS=0,DISP=D
// JOB ADDJOBS ADD JOBS TO POWER READER QUEUE
// EXEC DTRIINIT
   LOAD CICS2.Z
   LOAD PRTDUC2A.Z
   LOAD PRTDUC2B.Z
/*
/&
* $$ EOJ
```

Comment

The job below changes the share option definition from `SHAREOPTIONS(2)` to `SHAREOPTIONS(4)` for the Message Routing and the Host Transfer File.

Replace the ID of the VSE/VSAM user catalog if you do not use the default user catalog of z/VSE.

```
* $$ JOB JNM=SHARE4,CLASS=0,DISP=D
// JOB SHARE4 CHANGE SHAREOPTIONS
* PLEASE CLOSE FILES IESROUT AND INWFILE ON DBDCCICS
* AND ALSO ON ALL OTHER CICS PARTITIONS USING THE FILES.
*        CEMT SET FI(XXXXXXX) CLOSE
* A RETURN CODE OF 4 IS OK. IF THE INWFILE DOES NOT EXIST, RETURN
* CODE WILL BE 12.
* IF OTHER FILES SHOULD ALSO BE SHARED AMONG SYSTEMS CHANGE
* THE SHAREOPTIONS ACCORDINGLY.
// PAUSE
// EXEC IDCAMS
   ALTER VSE.MESSAGE.ROUTING.FILE.@I@ -
   SHAREOPTIONS(4)                    -
   CATALOG(VSESP.USER.CATALOG)
   /**/
```

Figure 56. Skeleton SKPREPC2 (Part 6 of 7)

Comment

Delete the two ALTER statements for the Host Transfer File if the available Workstation File Transfer Support is to be used only from your primary CICS Transaction Server or not at all.

If the Workstation File Transfer Support support is to be used from both CICS Transaction Servers, the share option of the Host Transfer File must be changed. This may slightly impact file transfer speed.

```
   ALTER VSE.MESSAGE.ROUTING.FILE.@D@ -
   SHAREOPTIONS(4)                    -
   CATALOG(VSESP.USER.CATALOG)
   /**/
   ALTER PC.HOST.TRANSFER.FILE.INDEX  -
   SHAREOPTIONS(4)                    -
   CATALOG(VSESP.USER.CATALOG)
   /**/
   ALTER PC.HOST.TRANSFER.FILE.DATA   -
   SHAREOPTIONS(4)                    -
   CATALOG(VSESP.USER.CATALOG)
/*
* OPEN FILES AGAIN
*        CEMT SET FI(XXXXXXX) OPEN
// PAUSE
/&
* $$ EOJ
* PLEASE CLOSE DFHCSD IN DBDCCICS
// PAUSE
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// EXEC DFHCSDUP,SIZE=600K        INIT AND LOAD CICS
  DELETE GROUP(FCTC2)
  DELETE LIST(VSELST2)
  ADD GROUP(VSETYPE) LIST(VSELST2)
  ADD GROUP(VSETERM) LIST(VSELST2)
  ADD GROUP(VSETERM1) LIST(VSELST2)
  APPEND LIST(DFHLIST) TO(VSELST2)
  ADD GROUP(DFHRCF) LIST(VSELST2)
  ADD GROUP(DFHCLNT) LIST(VSELST2)
  ADD GROUP(CICREXX) LIST(VSELST2)
  ADD GROUP(TCPIP) LIST(VSELST2)
  ADD GROUP(VSEAI62) LIST(VSELST2)
  ADD GROUP(EZA) LIST(VSELST2)
  ADD GROUP(DFH$WBSN) LIST(VSELST2)
* $$ SLI MEM=IESZFCT2.Z
  ADD GROUP(VSESPG) LIST(VSELST2)
  ADD GROUP(FCTC2) LIST(VSELST2)
  ADD GROUP(CEE) LIST(VSELST2)
  LIST ALL
/*
* PLEASE OPEN DFHCSD AGAIN
// PAUSE
/&
* $$ EOJ
```

Figure 56. Skeleton SKPREPC2 (Part 7 of 7)

Note

When you have completed both skeletons (SKCICS2 and SKPREPC2), continue with "Task 3: Modify CICS Control Tables" on page 193.

Chapter 9. Creating a CICS Coexistence Environment

In addition to the CICS Transaction Server, you can also install CICS/VSE 2.3 and create a CICS coexistence environment. This allows you to run applications that still require CICS/VSE 2.3 functions.

CICS/VSE 2.3 is shipped on the extended base tape and must be installed like an optional program.

Both CICS systems (CICS Transaction Server and CICS/VSE 2.3) can have their **own CSD (CICS System Definition)** file or **share a common CSD** file. The shared CSD always belongs to the CICS Transaction Server. For further details refer to the manual *z/VSE Planning*.

The support makes use of existing skeletons and provides:

SKCICSOL (for defining startup job)

SKPREPCO (for defining resources in case of **two separate** CSD files)

SKPREPSO (for defining resources in case of **one shared** CSD file)

A CICS/VSE 2.3 system under z/VSE does not include VSE/ICCF and can have one of the following relationships to the primary CICS, the CICS Transaction Server:

1. No communication with the primary CICS.
2. Communication to the primary CICS via *Multiregion Operation* (MRO). Both CICS systems run in the same processor.

Besides MRO, communication via ISC (Intersystem Communication) is also possible.

Note: There is no sign-on panel for CICS/VSE. Sign-on is performed by entering CSSN. To sign-off and return to the VTAM selection panel enter CSSF LOGOFF.

Before you begin with the installation process you should consult the manual *z/VSE Planning* under “Characteristics of a CICS Coexistence Environment”.

Installation Tasks for CICS/VSE 2.3

The tasks to be performed are described below. Follow the given sequence.

Task 1: Modify Predefined Environment

The following is assumed:

- You selected predefined **environment B** or **environment C** for initial installation.
- You run CICS/VSE 2.3 in **partition F4** (for storage key reasons: do not run a CICS TS system in F4). As shipped, z/VSE has the following partition values defined for F4:
  - In predefined environment B

    ```
    ALLOC F4=20M
    SIZE F4=2M
    ```

  - In predefined environment C

```
ALLOC F4=32M
SIZE F4=2M
```

Note that these are recommended average values. Depending on your applications they may not be sufficient.

For CICS/VSE 2.3 the name of the startup job is CICSOLD, the name of the corresponding skeleton is SKCICSOL.

Before you edit the skeletons, copy them first from VSE/ICCF library 59 to your primary library.

Increasing partition sizes may also mean that you have to increase the VSIZE to meet your total system requirements. The default VSIZE for predefined:
- Environment B is 264 MB.
- Environment C is 2 GB.

You can change the VIO and VSIZE values with the *Tailor IPL Procedure* dialog. Refer to "Tailoring the IPL Procedure" on page 8 for details.

Notes:
1. You may run CICS/VSE 2.3 in a **dynamic partition**. Predefined environments B and C support dynamic partitions.
2. Choosing a Z-class partition with 5MB would mean, for example, that you have to extend it by about 15MB to host your CICS/VSE 2.3.
3. The partition parameters and values for dynamic partitions are defined in the active dynamic class table. You can modify this table with the *Maintain Dynamic Partitions* dialog. For details refer to "Defining Dynamic Class Tables" on page 71.

Note:

Running CICS/VSE 2.3 in another static partition than F4 (or in a dynamic partition) requires changes in skeleton SKCICSOL and in procedure CPUVAR1. Affected are mainly statements which include partition-related information. Further details are provided with the skeleton descriptions.

Notes:
1. Before you submit skeletons SKCICSOL and SKPREPCO or SKPREPSO, ensure that you did run skeleton SKCOLD at least once. It updates procedure COLDJOBS which loads jobs into the VSE/POWER reader queue that are important for a COLD startup.
2. Ensure that you only modify the LIBDEF search chain in the partition running CICS/VSE and for the required jobs. You are strongly recommended *not* to define a CICS/VSE sub-library in the permanent LIBDEF chain. If you do so, serious problems might occur if you apply service to the CICS Transaction Server for VSE/ESA.

Task 2: Modify Skeletons Provided by z/VSE

Copy a skeleton first from VSE/ICCF library 59 to your primary library and modify the skeleton here. This ensures that you have a backup version of the original skeleton available.

Skeleton SKUSERBG

1. Locate the statement

```
* // PWR PRELEASE RDR,CICSOLD
```

2. Delete the asterisk and the blank in the first two columns. Refer also to Figure 12 on page 42.

   When the modified procedure is processed during system startup, the statement causes CICS/VSE 2.3 to be started.

Skeletons SKCICSOL and either SKPREPCO or SKPREPSO

Refer to “Skeleton SKCICSOL” on page 217, and either to “Skeleton SKPREPCO” on page 218 or “Skeleton SKPREPSO” on page 226 where the skeletons are shown in detail. Comments point out what might or should be modified and why a modification should be considered.

Modify the skeletons **but do not submit** them now.

Task 3: Modify CICS/VSE 2.3 Control Tables and Programs

The control tables that may need to be modified for CICS/VSE 2.3 are listed below. The related skeletons are available in VSE/ICCF library 59:

DFHSITCO	System Initialization Table
DFHDCTCO	Destination Control Table
DFHFCTCO	File Control Table for separate CSD
DFHPCTCO	Program Control Table, for separate CSD
DFHPCTSO	Program Control Table, for shared CSD
DFHPPTCO	Processing Program Table, for separate CSD
DFHPPTSO	Processing Program Table, for shared CSD
DFHPLTPO	Program List Table (shutdown)
DFHPLTSO	Program List Table (startup)
DFHTSTCO	Temporary Storage Table
DFHSNT	Sign-On Table (not suffixed, skeleton: DFHSNTCO)
DFHTCTCO	Terminal Control Table
DFHJCTCO	Journal Control Table
IESZATCO	Autoinstall program for terminals

Note on DFHPCTxx and DFHPPTxx

For each of the two tables, two skeletons are provided: CO and SO, where SO stands for shared. However, the table created has always the suffix **CO**. Both tables are to be used only if RDO is not used. To use the tables, you must compile the skeletons and SKPREPCO and SKPREPSO automatically migrate them into the CSD file.

Note on DFHTCTCO

The Terminal Control Table (DFHTCTCO) shipped for CICS/VSE 2.3 can be used unchanged, except if you define terminals yourself and do not use the autoinstall function. Differences of significance are indicated in subsequent paragraphs that discuss the various tables. These paragraphs point out possible modifications. Some changes have to be made to the tables to include or exclude CICS-to-CICS communication support via MRO. You may also add entries to meet local requirements.

CICS Journal Files

If your modified skeleton SKPREPCO includes DLBL and EXTENT statements for CICS **Journal Files**, you must provide the specifications for the corresponding Journal Control Tables (DFHJCTs). You can use skeletons DFHJCTCO and SKJOURO, stored in VSE/ICCF library 59, for that purpose. For a description of the DFHJCT macro, refer to the CICS/VSE manual *CICS Resource Definition Guide*.

System Initialization Table

This table, shipped as member (skeleton) **DFHSITCO**, **includes** significant changes.

– The table operands are set as follows:

```
DCT=CO      PPT=NO
FCT=CO      SUFFIX=CO
JCT=NO      TCT=CO
PCT=NO      TST=CO
```

  In case of a **separate** CICS/VSE CSD, change in DFHSITCO the suffix for the PCT and PPT entries to CO (if not yet set) if you want to use the tables later on. If you use RDO, do not change the setting for PCT and PPT and leave it as NO.

– The application name of CICS/VSE 2.3:

```
APPLID=OLDCICS
```

– The table activates the spool support of CICS/VSE 2.3 with the specification

```
SPOOL=(YES,B,A)
```

– The internal trace function is set to off, since CICS/VSE 2.3 is assumed to run applications in production mode:

```
TRACE=(800,OFF)
```

– The table, as shipped, does not activate the CICS-to-CICS communication via MRO. The related specifications are:

```
GRPLIST=VSELSTO
IRCSTRT=NO
ISC=NO
SYSIDNT=CICO
```

  If **MRO communication is to be used**, change the below listed entries (in DFHSITCO for CICS/VSE 2.3) from NO to YES:

```
IRCSTRT=YES
ISC=YES
```

  Ensure also that in table DFHSITSP for the primary CICS (the CICS Transaction Server), the below listed entries are set to YES (which is required for MRO communication):

```
IRCSTRT=YES
ISC=YES
```

Destination Control Table

This table is shipped as member (skeleton) **DFHDCTCO**; it includes no significant differences. Any TYPE=SDSCI entries that you need in addition are to be added immediately behind the box labeled

```
LOCAL ENTRIES FOR TYPE=SDSCI SHOULD BE PLACED BELOW THIS BOX
```

File Control Table

This table is shipped as member (skeleton) **DFHFCTCO**; it includes no significant differences. Any entries that you need in addition are to be added immediately behind the box labeled

```
LOCAL ENTRIES SHOULD BE PLACED BELOW THIS BOX
```

In case of a shared CSD file, you should define the CSD in the FCT as read-only file. You can do this by removing the comments in skeleton DFHFCTCO and by deactivating the COPY DFH$FCT statement.

Program Control Table

This table is shipped as members (skeletons) **DFHPCTCO and DFHPCTSO**, where SO stands for a shared CSD file. After processing , the table has always the suffix **CO**. For a shared CSD file, processing is different: DFHPCTSO must be assembled with CICS TS to be migrated into the CSD file.

Any entries that you need in addition are to be added immediately behind the box labeled

```
LOCAL ENTRIES SHOULD BE PLACED BELOW THIS BOX
```

To make your changes active, run the tables before job SKPREPCO or job SKPREPSO is executed.

Processing Program Table

This table is shipped as members (skeletons) **DFHPPTCO and DFHPPTSO**, where SO stands for a shared CSD file. After processing , the table has always the suffix **CO**. For a shared CSD file, processing is different: DFHPCTSO must be assembled with CICS TS to be migrated into the CSD file.

Any entries that you need in addition are to be added immediately behind the box labeled

```
LOCAL ENTRIES SHOULD BE PLACED BELOW THIS BOX
```

To make your changes active, run the tables before job SKPREPCO or job SKPREPSO is executed.

Program List Table

This table is shipped as member (skeleton) **DFHPLTPO** for shutdown and as member (skeleton) **DFHPLTSO** for startup. Both members need to be completed by adding user programs after the box labeled

```
LOCAL ENTRIES SHOULD BE PLACED BELOW THIS BOX
```

If no user programs are added, the compilation will fail. After adding programs, the related program list has to be activated in DFHSITCO by setting PLTPI=PO and PLTSD=SO.

Temporary Storage Table

This table is shipped as member (skeleton) **DFHTSTCO** and can be used unchanged. Any entries that you need in addition are to be added immediately behind the box labeled

```
LOCAL ENTRIES SHOULD BE PLACED BELOW THIS BOX
```

Sign-On Table

The sign-on table has no suffix (**DFHSNT**); the skeleton name is DFHSNTCO. As shipped, it includes the predefined users SYSA and OPER. Add user entries of your own as required for CICS/VSE 2.3.

Terminal Control Table

This table is shipped as member (skeleton) **DFHTSTCO**. CICS/VSE 2.3 assumes that terminals are to be defined with its autoinstall function. The DFHTCTCO therefore includes only console-related entries. If you need to use VTAM terminals and do not use the autoinstall function, you can use DFHTCTCO for defining them.

Journal Control Table

This table is shipped as member (skeleton) **DFHJCTCO**. If you want to use the table you must set JCT=CO in the system initialization table DFHSITCO. Refer also to "CICS Journal Files" on page 212.

Autoinstall Program IESZATCO

For terminals, CICS/VSE provides the autoinstall program IESZATCO. It is functionally equivalent to IESZATDX.

Task 4: Submit the Modified Skeletons

After having modified the skeletons as described under "Skeleton SKCICSOL" on page 217 and "Skeleton SKPREPCO" on page 218 or "Skeleton SKPREPSO" on page 226, submit the skeletons from the FULIST of your primary VSE/ICCF library. For SKCICSOL use option **7**.

If **MRO communication is to be used**, you should assign unique CICS terminal IDs (via program IESZATCO) for the terminals of CICS/VSE 2.3. Refer also to "Tailoring Autoinstall Terminals" on page 216.

Task 5: Definitions for MRO

For this task use the RDO (Resource Definition Online) function described in the manual *CICS TS Resource Definition Guide*. The function is a convenient means for setting up a communication path to the primary CICS Transaction Server and for defining terminals.

1. **Define MRO communication between CICS/VSE 2.3 and CICS Transaction Server**

   This requires the definition of a connection and an associated sessions definition as follows:

   For a **connection** definition, enter the RDO command

   ```
   CEDA DEFINE CONNECTION
   ```

   and provide specifications as listed below. Accept the defaults for the data-entry fields not listed here.

   Note: If you have a shared CSD, enter all CEDA commands from a CICS Transaction Server terminal. If each CICS system has its own CSD, you must enter the commands as required partly from a CICS Transaction Server and partly from a CICS/VSE terminal.

Panel Line	Specifications DBDCCICS Side	Specifications OLDCICS Side	Comment
Connection:	CICO	CIC1	SYSID in DFHSITC2

Panel Line	Specifications DBDCCICS Side	Specifications OLDCICS Side	Comment
Group:	VSEIRC1	VSEIRCO	
Netname:	OLDCICS	DBDCCICS	
ACcessmethod:	IRc	IRc	
Protocol:			Must be blank
AUToconnect:	Yes	Yes	

For the associated **sessions** definition, enter the RDO command

```
CEDA DEFINE SESSIONS
```

Provide the specifications listed below to have 10 send and receive sessions. Accept the defaults for the data-entry lines not listed here.

Panel Line	Specifications DBDCCICS Side	Specifications OLDCICS Side	Comment
Sessions:	CICSSO	CICSS1	arbitrary
Group:	VSEIRC1	VSEIRCO	
Connection:	CICO	CIC1	SYSID in DFHSITC2
Protocol:	LU61	LU61	
RECEIVEPfx:	TR	PR	
RECEIVECount:	010	010	
SENDPfx:	TS	PS	
SENDCount:	010	010	
SENDSize:	4096	4096	See Note a below
RECEIVESize:	4096	4096	See Note a below
OPERRsl:	0	0	See Note b below
OPERSecurity:	1	1	See Note b below
AUToconnect:	Yes	Yes	
INservice:	Yes	Yes	
RELreq:	Yes	Yes	
Discreq:	Yes	Yes	

Notes:

a. A general recommendation: certain CICS applications may require specific values to be specified. Check the applicable manuals.

b. This is the default: gives the terminal operator access to unprotected resources only.

If MRO is to be used for PRODCICS (second CICS TS), appropriate definitions may also be added as described under “Task 5: Definitions for MRO” on page 195 or “Task 3: Modify CICS Control Tables” on page 193.

2. **Define terminals**

Use the autoinstall program (IESZATCO) of CICS/VSE 2.3 to define the terminals that are to be supported by CICS/VSE.

Enter RDO commands from a CICS/VSE terminal if you have separate CSD files (otherwise from a CICS Transaction Server terminal) as follows:

```
CEDA ADD GROUP(VSEIRCO) LIST(VSELSTO)
```

This statement defines a new CSD list, VSELSTO, for CICS/VSE 2.3. Terminal definitions specified in group VSETERM1 are also copied into VSELSTO. The statement causes all of list VSELIST to be copied into the new CSD list, and the new group VSEIRCO (which you had defined via CEDA DEFINE SESSIONS), to be added to the list.

```
CEDA ADD GROUP(VSEIRC1) LIST(VSELIST)
CEDA INSTALL GROUP(VSEIRC1)
```

The above statements add and install the new CSD group, `VSEIRC1`, which you had defined via CEDA DEFINE SESSIONS for your primary CICS Transaction Server. Enter them from a CICS Transaction Server terminal.

After successful completion of the above procedure, the required definitions for CICS/VSE 2.3 are complete. At the next startup of your z/VSE system, CICS/VSE 2.3 will be available.

Tailoring Autoinstall Terminals

For using unique CICS terminal IDs, you have to perform the following steps:

- In the *Hardware Configuration* dialog select option 3 for logical unit (further processing) for autoinstall terminals.
- If the entry for CICS TERM ID is displayed, use option 6 and delete the entry for TERM ID (CICS).

The TERM IDs (such as A001, A002 and so on) are used for the first and a second CICS.

To prevent the use of duplicated CICS TERM IDs for CICS/VSE 2.3, do the following:

- Access and copy the autoinstall program member IESZATCO in VSE/ICCF library 59 into your own VSE/ICCF library.
- Locate the field **PREFIX DC** ’**A**’.
- Change the letter ’A’ to any other letter (for example, ’B’) which is to be used as prefix for the CICS TERM ID.
- Submit the modified member IESZATCO.

Considerations for Problem Solving

As shipped, the startup job stream (skeleton SKCICSOL) defines and allocates an AUXTRACE file. Yet, tracing the flow of transactions through the system is not activated automatically. For how to use AUXTRACE, refer to the *CICS Problem Determination Guide*. Skeleton DFHAUXOL, which is stored in VSE/ICCF library 59, provides a job stream for analyzing AUXTRACE data. The label information in the job stream is set for CICS/VSE 2.3 (OLDCICS).

z/VSE provides skeleton SKCIDUMP to print formatted CICS/VSE dumps.

You can use the dialog *Storage Dump Management* to process dumps created either by CICS TS or CICS/VSE 2.3.

Skeletons for CICS/VSE 2.3

This section lists and describes the CICS/VSE 2.3 skeletons:

SKCICSOL
SKPREPCO
SKPREPSO
SKJOURO

If you want to use a **shared** CSD file, you must use skeleton SKPREPSO instead of SKPREPCO.

These skeletons are shipped as members of VSE/ICCF library 59. The skeletons as listed in this section include comments that you may find helpful when modifying them.

Also note that names of resources (such as volumes) that you might want or have to change are highlighted in bold.

Skeleton SKCICSOL

This is the startup procedure for CICS/VSE 2.3. You can submit this skeleton unchanged if CICS/VSE 2.3 runs in partition F4. Else, change the highlighted specifications in this skeleton accordingly and also the XAPPLF4 definition in the CPUVAR1 procedure (via DTRSETP).

The first loading of CICSOLD is done via skeleton SKPREPCO. If there is a need to load CICSOLD again, you have to remove the asterisk (*) in front of the command `EXEC PROC=LDCICSOL` to activate the loading into the VSE/POWER reader queue.

After you have modified skeleton SKCICSOL, enter the following command from the editor's command line **before** you file the skeleton:

```
@DTRSEXIT
```

This command calls a macro that deletes specific comments from the skeleton.

```
* $$ JOB JNM=CATCICSO,DISP=D,CLASS=0
// JOB CATCICSO                  CATALOG CICSOLD AND LDCICSOL
// EXEC LIBR,PARM='MSHP'
   ACCESS S=IJSYSRS.SYSLIB
   CATALOG CICSOLD.Z REPLACE=YES
$$$$ JOB JNM=CICSOLD,DISP=L,CLASS=4,EOJMSG=YES
$$$$ LST CLASS=A,DISP=D,RBS=100
// JOB CICSOLD         STARTUP OF COEXISTING  CICS 2.3 WITHOUT II
// OPTION SADUMP=5
// LIBDEF *,SEARCH=(PRD2.CICSOLDP,PRD2.CONFIG,PRD2.PROD,              X
               PRD2.SCEEBASE,PRD2.DBASE,PRD1.BASE)
// SETPARM XNCPU=''
// SETPARM XMODEF4=''
// SETPARM XAPPLF4=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF4=ACTIVE' **F4 ASSUMED
$$/*
// EXEC PROC=CPUVAR&XNCPU,XMODEF4,XAPPLF4        **F4 ASSUMED
// SETPFIX LIMIT=256K
*  ASSGN SYS029,DISK,VOL=SYSWK1,SHR    IF JOURNALLING
// EXEC PROC=DTRCICSO                  LABELS FOR CICS FILES
// IF XMODEF4 = COLD THEN                               **F4 ASSUMED
// GOTO COLD
```

Comment

If you use another name than CICSOLD, you must also update procedures such as USERBG and COLDJOBS (via skeletons SKUSERBG and SKCOLD).

```
// EXEC DFHSIP,SIZE=6344K,PARM='SIT=CO,APPLID=&XAPPLF4.,$END',DSPACE=2M
$$/*
// GOTO END
/. COLD
// EXEC DFHSIP,SIZE=6344K,PARM='SIT=CO,APPLID=&XAPPLF4.,START=COLD,$END'*
               ,DSPACE=2M
$$/*
/. END
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF4=INACTIVE'
$$/*
$$/&
$$$$ EOJ
/+
CATALOG  LDCICSOL.PROC     REPLACE=YES DATA=YES
// EXEC DTRIINIT
   LOAD CICSOLD.Z
/*
/+
/*
// EXEC PROC=LDCICSOL       LOAD CICSOLD INTO RDR QUEUE
/*
/&
* $$ EOJ
```

Figure 57. Skeleton SKCICSOL (Starting up CICS/VSE 2.3 in Partition F4)

The skeleton includes additional $$ characters. They are needed to mask off VSE/POWER JECL statements. Program DTRIINIT, described in the manual *z/VSE System Utilities* under “DTRIINIT Utility”, replaces the $$ characters with VSE/POWER JECL statements for cataloging.

Skeleton SKPREPCO

The skeleton defines the resources for a CICS/VSE 2.3 system and catalogs the required label information, where both CICS systems have their own CSD file.

After the CICS tables have been modified and compiled, SKPREPCO will update the CICS/VSE CSD file using the generated phases DFHPCTCO and DFHPPTCO. All other tables are kept separate and are not stored in the CSD.

Before you submit SKPREPCO for processing, the following skeletons and tables must be submitted for processing first:
1. SKCICSOL (startup)
2. SKUSERBG (if you have a need to change it)
3. Modified and compiled CICS tables except DFHSNTCO.

In DFHSITCO, the PCT and PPT entries are set as follows:

```
PCT=NO
PPT=NO
```

These entries should not be changed, except if you want to continue to use these tables (instead of the CSD file). Then specify ″CO″ instead of ″NO″.

After you have submitted SKPREPCO modify and submit table DFHSNTCO.

```
* $$ JOB JNM=VSAMDEFO,DISP=D,CLASS=0
// JOB VSAMDEFO - DEFINE VSAM CLUSTERS FOR OLD CICS
* **********************************************
*  DEFINE AND INITIALIZE VSAM FILES FOR CICSOLD*
* **********************************************
// EXEC IDCAMS,SIZE=AUTO
 /*                                                           */
 /*  DELETE VSAM FILES                                        */
 /*                                                           */
```

Comment

The delete job below ensures that no catalog entries with identical file IDs exist.

You might have files with an ID identical to the ones specified in the job and also under control of the specified user catalog. If you need those files further on (to operate with three CICS systems, for example), rename the IDs in the skeleton. Suggested approach: change `CICSO` to `CICSOB`.

If you do not use the default user catalog, do a global change for the catalog name, **`VSESPUC`**, and the catalog ID, **`VSESP.USER.CATALOG`**. The occurrences of these specifications in the skeleton are highlighted.

```
  DELETE (CICSO.CSD) CL NOERASE PURGE      -
  CATALOG(VSESP.USER.CATALOG)
  DELETE (CICSO.DFHTEMP) CL NOERASE PURGE  -
  CATALOG(VSESP.USER.CATALOG)
  DELETE (CICSO.TD.INTRA) CL NOERASE PURGE  -
  CATALOG(VSESP.USER.CATALOG)
  DELETE (CICSO.AUTO.STATS.A) CL NOERASE PURGE -
```

Figure 58. Skeleton SKPREPCO (Preparation Jobs for CICS/VSE 2.3) (Part 1 of 7)

Comment

The required VSE/VSAM files are defined to reside on the SYSWK1 volume with DOSRES specified as the secondary volume for allocations. If you plan to have the files allocated in VSE/VSAM space on different volumes, change the volume names accordingly.

```
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICSO.AUTO.STATS.B) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICSO.RSD) CL NOERASE PURGE  -
    CATALOG(VSESP.USER.CATALOG)
    SET MAXCC = 0
/*                                                              */
/*  DEFINE VSAM FILES                                           */
/*                                                              */
DEF CLUSTER(NAME(CICSO.DFHTEMP)          -
    VOL(SYSWK1 DOSRES)                   -
    RECORDS (100)                        -
    RECORDSIZE (4089 4089)               -
    CISZ   (4096)                        -
    NONINDEXED                           -
    SHR(2))                              -
    DATA(NAME(CICSO.DFHTEMP.ESDS))       -
    CATALOG(VSESP.USER.CATALOG)
/*                                                              */
DEF CLUSTER(NAME(CICSO.TD.INTRA)         -
    VOL(SYSWK1 DOSRES)                   -
    RECORDS (100)                        -
    RECORDSIZE (4089 4089)               -
    CISZ   (4096)                        -
    NONINDEXED                           -
    SHR(2))                              -
    DATA(NAME(CICSO.TD.INTRA.ESDS))      -
    CATALOG(VSESP.USER.CATALOG)
/*                                                              */
DEF CLUSTER (NAME(CICSO.AUTO.STATS.A)    -
    NONINDEXED                           -
    RECORDFORMAT(VARUNB)                 -
    RECORDSIZE(304,304)                  -
    FOR(0)                               -
    REUSE                                -
    RECORDS(20,10)                       -
    VOLUME(SYSWK1 DOSRES))               -
    DATA(NAME(CICSO.AUTO.STATS.A.ESDS))  -
    CATALOG(VSESP.USER.CATALOG)
/*                                                              */
DEF CLUSTER (NAME(CICSO.AUTO.STATS.B)    -
    NONINDEXED                           -
    RECORDFORMAT(VARUNB)                 -
    RECORDSIZE(304,304)                  -
    FOR(0)                               -
    REUSE                                -
    RECORDS(20,10)                       -
    VOLUME(SYSWK1 DOSRES))               -
    DATA(NAME(CICSO.AUTO.STATS.B.ESDS))  -
    CATALOG(VSESP.USER.CATALOG)
/*                                                              */
```

Figure 58. Skeleton SKPREPCO (Preparation Jobs for CICS/VSE 2.3) (Part 2 of 7)

```
 DEF CLUSTER(NAME(CICSO.RSD)               -
     INDEXED                               -
     RECORDSIZE (2000 2000)                -
     RECORDS (250 100)                     -
     KEYS   (22 0)                         -
     FREESPACE (20 20)                     -
     SHR(2)                                -
     VOL(SYSWK1 DOSRES))                   -
     DATA(NAME(CICSO.RSD.@D@))             -
     INDEX (NAME (CICSO.RSD.@I@))          -
     CATALOG(VSESP.USER.CATALOG)
 DEF CLUSTER(NAME(CICSO.CSD)               -
     INDEXED                               -
     RECORDSIZE (100 500)                  -
     RECORDS (4000 1000)                   -
     KEYS   (22 0)                         -
     FREESPACE (0 0)                       -
     SHR(2)                                -
     VOL(SYSWK1 DOSRES))                          -
     DATA(NAME(CICSO.CSD.@D@))             -
     INDEX (NAME (CICSO.CSD.@I@))          -
     CATALOG(VSESP.USER.CATALOG)
 /*                                                             */
 /**************************************************************/
 /*  STANDARD LAYOUT DEFINES SEPARATE CSD FILES FOR NEW AND OLD   */
 /*  CICS. THIS ALLOWS CHANGES ON THE OLD CICS USING CEDA.        */
 /**************************************************************/
/*
// IF $RC > 0 THEN
CANCEL
*
*    INITIALIZE THE CICSOLD RESTART DATA SET
*
// DLBL DFHRSD,'CICSO.RSD',0,VSAM,CAT=VSESPUC
// EXEC IDCAMS,SIZE=AUTO
   REPRO INFILE   -
        (SYSIPT                 -
         ENVIRONMENT            -
          (RECORDFORMAT (FIXUNB) -
           BLOCKSIZE(80)        -
           RECORDSIZE (80)))    -
        OUTFILE (DFHRSD)
ACTL 0001
/*
/&
* $$ EOJ
```

Figure 58. Skeleton SKPREPCO (Preparation Jobs for CICS/VSE 2.3) (Part 3 of 7)

```
* $$ JOB JNM=DTRCICSO,DISP=D,CLASS=0
// JOB DTRCICSO - DEFINE LABELS FOR COEXISTENT CICS
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG DTRCICSO.PROC  D=YES  R=YES  EOD=/+
// ASSGN SYS018,DISK,VOL=SYSWK1,SHR
// DLBL DFHDMPA,'CICSO.DUMPA',0,VSAM,                                  X
               CAT=VSESPUC,RECSIZE=7200,                               X
               DISP=(NEW,KEEP),RECORDS=(300,0)
// DLBL DFHDMPB,'CICSO.DUMPB',0,VSAM,                                  X
               CAT=VSESPUC,RECSIZE=7200,                               X
               DISP=(NEW,KEEP),RECORDS=(100,0)
// DLBL DFHAUXT,'CICSO.AUXTRACE',0,VSAM,                               X
               CAT=VSESPUC,RECSIZE=4096,                               X
               DISP=(NEW,KEEP),RECORDS=(400,0)
// DLBL DFHTEMP,'CICSO.DFHTEMP',0,VSAM,                                X
               CAT=VSESPUC
// DLBL DFHNTRA,'CICSO.TD.INTRA',0,VSAM,                               X
               CAT=VSESPUC
// DLBL DFHSTM,'CICSO.AUTO.STATS.A',0,VSAM,                            X
               CAT=VSESPUC,                                            X
               DISP=(,DELETE)
// DLBL DFHSTN,'CICSO.AUTO.STATS.B',0,VSAM,                            X
               CAT=VSESPUC,                                            X
               DISP=(,DELETE)
// DLBL DFHRSD,'CICSO.RSD',0,VSAM,                                     X
               CAT=VSESPUC
// DLBL DFHCSD,'CICSO.CSD',0,VSAM,                                     X
               CAT=VSESPUC
/+
  CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
  COPY DTRCICSO.PROC  REPLACE=YES
  AC S=PRD2.CICSOLDP
  REN DFHSNT.PHASE:DFHSNTDY.PHASE
/*
/&
* $$ EOJ
```

Figure 58. Skeleton SKPREPCO (Preparation Jobs for CICS/VSE 2.3) (Part 4 of 7)

Comment

If you plan to define and format one or more CICS journal files, insert the required DLBL and EXTENT statements at this point. Member SKJOURO of the VSE/ICCF library 59 includes suitable sample statements for all types of the supported IBM disk devices. For further details about defining journal files, refer to “CICS Journal Files” on page 212.

A separate CSD file is newly created for CICS/VSE. The CICS/VSE provided DFHSNT is renamed to DFHSNTDY.

The SETPARM values specified in the job below assume that CICS/VSE 2.3 will run in partition F4. If this is your intent, no partition-related specifications need be changed. However, if CICS/VSE 2.3 is to run in another partition, F5 for example, change the SETPARM values as shown:

Current Specification	Change To
XPARTCO='F4'	XPARTCO='F5'
XUSEF4='CI'	XUSEF5='CI'

In addition, you must change in CPUVAR1 (via DTRSETP) the following entry:

Current Specification	Change To
XAPPLF4='OLDCICS'	XAPPLF5='OLDCICS'

```
* $$ JOB JNM=GLOBVAR,DISP=D,CLASS=0
// JOB GLOBVAR - DEFINE GLOBAL VARIABLES
// SETPARM XNCPU=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU'
SET XPARTCO='F4'
SET XUSEF4='CI'
SET XAPPLF4='OLDCICS'
/*
/&
* $$ EOJ
```

Comment

The following job replaces the currently used startup procedure USERBG for the BG partition. In the SLI statement, replace the library number 59 by the number of your primary library (assuming that you had applied the changes to SKUSERBG in your primary library). Jobs PRTDUCOA and PRTDUCOB are forprinting the dump data sets of CICS/VSE 2.3.

If your currently used startup procedure includes changes of your own, ensure that you **apply the same changes** to skeleton SKUSERBG **before you submit skeleton SKPREPCO**.

Figure 58. Skeleton SKPREPCO (Preparation Jobs for CICS/VSE 2.3) (Part 5 of 7)

```
* $$ JOB JNM=COPYUBG,DISP=D,CLASS=0
// JOB COPYUBG - COPY SKUSERBG FROM ICCF LIBRARY
// EXEC LIBR,PARM='MSHP'
   ACCESS S=IJSYSRS.SYSLIB
* $$ SLI ICCF=(SKUSERBG),LIB=(59)
/*
/&
* $$ EOJ
* $$ JOB JNM=ADDJOBS,CLASS=0,DISP=D
// JOB ADDJOBS ADD JOBS TO POWER READER QUEUE
// EXEC DTRIINIT
   LOAD CICSOLD.Z
   LOAD PRTDUCOA.Z
   LOAD PRTDUCOB.Z
/*
/&
* $$ EOJ
```

Comment

The job below initializes the separate CSD file for CICS/VSE 2.3.

Replace the ID of the VSE/VSAM user catalog if you do not use the default user catalog of z/VSE.

```
* $$ JOB JNM=DFHCSDOL,CLASS=0,DISP=D
// JOB DFHCSDOL UPGRADE THE CSD FILE FOR COEXISTENT CICS
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD2.CICSOLDP,PRD2.SCEEBASE,PRD1.BASE)
// DLBL DFHCSD,'CICSO.CSD',0,VSAM,                                     X
               CAT=VSESPUC
// EXEC DFHCSDUP,SIZE=600K        INIT AND LOAD CICS CSD VSAM FILE
 INITIALIZE
 MIGRATE TABLE(DFHPPTCO) TOGROUP(VSESPO)
 MIGRATE TABLE(DFHPCTCO) TOGROUP(VSESPT)
 UPGRADE USING(IESMODEL)
 UPGRADE USING(DFHCU230)
 UPGRADE USING(DFHCU23F)
 COPY GROUP(VSESPT) TO(VSESPO)
 DELETE GROUP(VSESPT)
 APPEND LIST(DFHLIST) TO(VSELSTO)
 ADD GROUP(VSETYPE) LIST(VSELSTO)
 ADD GROUP(VSETERM) LIST(VSELSTO)
 ADD GROUP(VSETERM1) LIST(VSELSTO)
 ADD GROUP(VSESPO) LIST(VSELSTO)
* $$ SLI MEM=CEECCSD.Z,S=(PRD2.SCEEBASE)
* $$ SLI MEM=IBMCCSD.Z,S=(PRD2.SCEEBASE)
* $$ SLI MEM=IGZCCSD.Z,S=(PRD2.SCEECICS)
* $$ SLI MEM=EDCCCSD.Z,S=(PRD2.SCEEBASE)
* $$ SLI MEM=EDCUCSD.Z,S=(PRD2.SCEEBASE)
 ADD GROUP(CEE) LIST(VSELSTO)
* $$ SLI MEM=IPNCSD.Z,S=(PRD1.BASE)
 ADD GROUP(TCPIP) LIST(VSELSTO)
   LIST ALL
/*
/&
* $$ EOJ
```

Figure 58. Skeleton SKPREPCO (Preparation Jobs for CICS/VSE 2.3) (Part 6 of 7)

Comment

DUMPEXT ensures that the CICS Dump Analysis Routine (DFHDAP) is known to the system.

```
* $$ JOB JNM=DUMPEXT,CLASS=0,DISP=D
// JOB DUMPEXT EXTEND EXTERNAL ROUTINE FILE
// LIBDEF *,SEARCH=(PRD2.CICSOLDP,PRD1.BASE)
// UPSI 1
// SETPARM DASD=''                  INITIALIZE DASD PARAMETER
// SETPARM XNCPU=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC PROC=CPUVAR&XNCPU,DASD       GET DASD VALUE
// EXEC PROC=DTRINFOA            GET INFO ANALYSIS ASSGNS
* ------------------------------------------------------------
* YOU WILL GET MESSAGE
* 4433D EQUAL FILE ID IN VTOC  BLNXTRN  SYS017=XXX  SYSWK1
* INFO.ANALYSIS.EXT.RTNS.FILE
* ENTER 'DELETE'
* ------------------------------------------------------------
// IF DASD EQ GFBA THEN
// GOTO INFBA
/. INCKD                INITIALIZE EXTERNAL ROUTINES FILE ON CKD
// EXEC DITTO
$$DITTO CSQ BLKFACTOR=1,FILEOUT=BLNXTRN
ANEXIT DFHDP410 CICS DUMP ANALYZER
ANEXIT DFHDAP   CICS 2.3 DUMP ANALYZER
ANEXIT IJBXCSMG ANALYSE CONSOLE BUFFER
ANEXIT IJBXDBUG ANALYSE STANDALONE DUMP ROUTINE
ANEXIT IJBXSDA  SDAID BUFFER FORMATTING ROUTINE
/*
$$DITTO EOJ
/*
// GOTO EXIT
/. INFBA                INITIALIZE EXTERNAL ROUTINES FILE ON FBA
// EXEC DITTO
$$DITTO CSQ BLKFACTOR=1,FILEOUT=BLNXTRN,CISIZE=512
ANEXIT DFHDP410 CICS DUMP ANALYZER
ANEXIT DFHDAP   CICS 2.3 DUMP ANALYZER
ANEXIT IJBXCSMG ANALYSE CONSOLE BUFFER
ANEXIT IJBXDBUG ANALYSE STANDALONE DUMP ROUTINE
ANEXIT IJBXSDA  SDAID BUFFER FORMATTING ROUTINE
/*
$$DITTO EOJ
/*
/. EXIT                END OF INFO/ ANALYSIS FILE INITIALIZE
/&
* $$ EOJ
```

Figure 58. Skeleton SKPREPCO (Preparation Jobs for CICS/VSE 2.3) (Part 7 of 7)

Skeleton SKPREPSO

The skeleton defines the resources for a CICS/VSE 2.3 system and catalogs the required label information, where the two CICS systems share the CSD file of the CICS Transaction Server.

After the CICS tables have been modified and compiled, SKPREPSO will update the CICS/VSE CSD file using the generated phases DFHPCTSO and DFHPPTSO. (All other tables are kept separate and are not stored in the CSD.)

Before you submit SKPREPSO for processing, the following skeletons and tables must be submitted for processing first:

1. SKCICSOL (startup)
2. SKUSERBG (if you have a need to change it)
3. Modified and compiled CICS tables except DFHSNTSO.

In DFHSITSO, verify that the PCT and PPT entries are as follows:

```
PCT=NO
PPT=NO
```

After you have submitted SKPREPSO modify and submit table DFHSNTSO.

```
$$ JOB JNM=VSAMDEFO,DISP=D,CLASS=0
// JOB VSAMDEFO - DEFINE VSAM CLUSTERS FOR OLD CICS
* *****************************************************************  C
* *                                                               *  C
* * - - - - - - - PREPARATION JOBS FOR FOR OLDCICS CICS 2.3  - - -   *  C
* * - - - - - - - - - - - FOR CSD FILE SHARED WITH CICS TS - - - -   *  C
* *                                                               *
* ********************************************
*  DEFINE AND INITIALIZE VSAM FILES FOR CICSOLD*
* ********************************************
// EXEC IDCAMS,SIZE=AUTO
 /*                                                             */
 /*  DELETE VSAM FILES                                          */
 /*                                                             */
```

Figure 59. Skeleton SKPREPSO (Preparation Jobs for CICS/VSE 2.3) (Part 1 of 7)

Comment

The delete statements below ensure that no catalog entries with identical file IDs exist.

You might have files with an ID identical to the ones specified in the job and also under control of the specified user catalog. If you need those files further on (to operate with three CICS systems, for example), rename the IDs in the skeleton. Suggested approach: change `CICSO` to `CICSOB`.

If you do not use the default user catalog, do a global change for the catalog name, **VSESPUC**, and the catalog ID, **VSESP.USER.CATALOG**. The occurrences of these specifications in the skeleton are highlighted.

```
DELETE (CICSO.DFHTEMP) CL NOERASE PURGE  -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICSO.TD.INTRA) CL NOERASE PURGE  -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICSO.AUTO.STATS.A) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICSO.AUTO.STATS.B) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICSO.RSD) CL NOERASE PURGE  -
    CATALOG(VSESP.USER.CATALOG)
    SET MAXCC = 0
```

Comment

The required VSE/VSAM files are defined to reside on the SYSWK1 volume with DOSRES specified as the secondary volume for allocations. If you plan to have the files allocated in VSE/VSAM space on different volumes, change the volume names accordingly.

```
/*                                                             */
/*  DEFINE VSAM FILES                                          */
/*                                                             */
DEF CLUSTER(NAME(CICSO.DFHTEMP)           -
    VOL(SYSWK1 DOSRES)                    -
    RECORDS (100)                         -
    RECORDSIZE (4089 4089)                -
    CISZ   (4096)                         -
    NONINDEXED                            -
    SHR(2))                               -
    DATA(NAME(CICSO.DFHTEMP.ESDS))        -
    CATALOG(VSESP.USER.CATALOG)
/*                                                             */
```

Figure 59. Skeleton SKPREPSO (Preparation Jobs for CICS/VSE 2.3) (Part 2 of 7)

```
 DEF CLUSTER(NAME(CICSO.TD.INTRA)          -
     VOL(SYSWK1 DOSRES)                    -
     RECORDS (100)                         -
     RECORDSIZE (4089 4089)                -
     CISZ   (4096)                         -
     NONINDEXED                            -
     SHR(2))                               -
     DATA(NAME(CICSO.TD.INTRA.ESDS))       -
     CATALOG(VSESP.USER.CATALOG)
 /*                                                                   */
 DEF CLUSTER (NAME(CICSO.AUTO.STATS.A)     -
     NONINDEXED                            -
     RECORDFORMAT(VARUNB)                  -
     RECORDSIZE(304,304)                   -
     FOR(0)                                -
     REUSE                                 -
     RECORDS(20,10)                        -
     VOLUME(SYSWK1 DOSRES))                -
     DATA(NAME(CICSO.AUTO.STATS.A.ESDS))   -
     CATALOG(VSESP.USER.CATALOG)
/*                                                                    */
 DEF CLUSTER (NAME(CICSO.AUTO.STATS.B)     -
     NONINDEXED                            -
     RECORDFORMAT(VARUNB)                  -
     RECORDSIZE(304,304)                   -
     FOR(0)                                -
     REUSE                                 -
     RECORDS(20,10)                        -
     VOLUME(SYSWK1 DOSRES))                -
     DATA(NAME(CICSO.AUTO.STATS.B.ESDS))   -
     CATALOG(VSESP.USER.CATALOG)
 /*                                                                   */
 DEF CLUSTER(NAME(CICSO.RSD)               -
     INDEXED                               -
     RECORDSIZE (2000 2000)                -
     RECORDS (250 100)                     -
     KEYS   (22 0)                         -
     FREESPACE (20 20)                     -
     SHR(2)                                -
     VOL(SYSWK1 DOSRES))                   -
     DATA(NAME(CICSO.RSD.@D@))             -
     INDEX (NAME (CICSO.RSD.@I@))          -
     CATALOG(VSESP.USER.CATALOG)
 /*                                                                   */
```

Figure 59. Skeleton SKPREPSO (Preparation Jobs for CICS/VSE 2.3) (Part 3 of 7)

```
 /*****************************************************************/
 /*  FOLLOWING FILES ARE SHARED WITH CICSICCF:                    */
 /*      -CICS.CSD                                                */
 /*****************************************************************/
/*
// IF $RC > 0 THEN
CANCEL
*
*    INITIALIZE THE CICSOLD RESTART DATA SET
*
// DLBL DFHRSD,'CICSO.RSD',0,VSAM,CAT=VSESPUC
// EXEC IDCAMS,SIZE=AUTO
   REPRO INFILE   -
        (SYSIPT                  -
         ENVIRONMENT             -
          (RECORDFORMAT (FIXUNB) -
           BLOCKSIZE(80)         -
           RECORDSIZE (80)))     -
        OUTFILE (DFHRSD)
ACTL 0001
/*
/&
$$ EOJ
$$ JOB JNM=DTRCICSO,DISP=D,CLASS=0
// JOB DTRCICSO - DEFINE LABELS FOR COEXISTENT CICS
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG DTRCICSO.PROC  D=YES  R=YES  EOD=/+
// ASSGN SYS018,DISK,VOL=SYSWK1,SHR
// DLBL DFHDMPA,'CICSO.DUMPA',0,VSAM,                                  X
               CAT=VSESPUC,RECSIZE=7200,                               X
               DISP=(NEW,KEEP),RECORDS=(300,0)
// DLBL DFHDMPB,'CICSO.DUMPB',0,VSAM,                                  X
               CAT=VSESPUC,RECSIZE=7200,                               X
               DISP=(NEW,KEEP),RECORDS=(100,0)
// DLBL DFHAUXT,'CICSO.AUXTRACE',0,VSAM,                               X
               CAT=VSESPUC,RECSIZE=4096,                               X
               DISP=(NEW,KEEP),RECORDS=(400,0)
// DLBL DFHTEMP,'CICSO.DFHTEMP',0,VSAM,                                X
               CAT=VSESPUC
// DLBL DFHNTRA,'CICSO.TD.INTRA',0,VSAM,                               X
               CAT=VSESPUC
// DLBL DFHSTM,'CICSO.AUTO.STATS.A',0,VSAM,                            X
               CAT=VSESPUC,                                            X
               DISP=(,DELETE)
// DLBL DFHSTN,'CICSO.AUTO.STATS.B',0,VSAM,                            X
               CAT=VSESPUC,                                            X
               DISP=(,DELETE)
// DLBL DFHRSD,'CICSO.RSD',0,VSAM,                                     X
               CAT=VSESPUC
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
  COPY DTRCICSO.PROC  REPLACE=YES
  AC S=PRD2.CICSOLDP
  REN DFHSNT.PHASE:DFHSNTDY.PHASE
/*
/&
$$ EOJ
```

Figure 59. Skeleton SKPREPSO (Preparation Jobs for CICS/VSE 2.3) (Part 4 of 7)

Comment

If you plan to define and format one or more CICS journal files, insert the required DLBL and EXTENT statements at this point. Member SKJOURO of the VSE/ICCF library 59 includes suitable sample statements for all types of the supported IBM disk devices. For further details about defining journal files, refer to "CICS Journal Files" on page 212.

No CSD file is created for CICS/VSE since the two CICS systems share the CSD file of the CICS Transaction Server. The CICS/VSE provided SNT is renamed to DFHSNTDY.

The SETPARM values specified in the job below assume that CICS/VSE 2.3 will run in partition F4. If this is your intent, no partition-related specifications need be changed. However, if CICS/VSE 2.3 is to run in another partition, F5 for example, change the SETPARM values as shown:

```
    Current Specification       Change To
      XPARTCO='F4'             XPARTCO='F5'
      XUSEF4='CI'              XUSEF5='CI'
```

In addition, you must modify in CPUVAR1 (via DTRSETP) the following entry:

```
    Current Specification       Change To
      XAPPLF4='OLDCICS'        XAPPLF5='OLDCICS'
```

```
$$ JOB JNM=GLOBVAR,DISP=D,CLASS=0
// JOB GLOBVAR - DEFINE GLOBAL VARIABLES
// SETPARM XNCPU=' '
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU'
   SET XPARTCO='F4'
   SET XUSEF4='CI'
   SET XAPPLF4='OLDCICS'
/*
/&
$$ EOJ
```

Comment

The following job replaces the currently used startup procedure USERBG for the BG partition. In the SLI statement, replace the library number 59 by the number of your primary library (assuming that you had applied the changes to SKUSERBG in your primary library). Jobs PRTDUCOA and PRTDUCOB are forprinting the dump data sets of CICS/VSE 2.3.

If your currently used startup procedure includes changes of your own, ensure that you **apply the same changes** to skeleton SKUSERBG **before you submit skeleton SKPREPCO**.

Figure 59. Skeleton SKPREPSO (Preparation Jobs for CICS/VSE 2.3) (Part 5 of 7)

```
$$ JOB JNM=COPYUBG,DISP=D,CLASS=0
// JOB COPYUBG - COPY SKUSERBG FROM ICCF LIBRARY
// EXEC LIBR,PARM='MSHP'
   ACCESS S=IJSYSRS.SYSLIB
 $$ SLI ICCF=(SKUSERBG),LIB=(59)
 /*
 /&
 $$EOJ
$$ JOB JNM=ADDJOBS,CLASS=0,DISP=D
 // JOB ADDJOBS ADD JOBS TO POWER READER QUEUE
 // EXEC DTRIINIT
    LOAD CICSOLD.Z
    LOAD PRTDUCOA.Z
    LOAD PRTDUCOB.Z
 /*
 /&
 $$ EOJ
 $$ JOB JNM=DFHCSDSO,CLASS=0,DISP=D
 // JOB DFHCSDSO UPGRADE THE COMMON CSD FILE FOR COEXISTENT CICS
 // LIBDEF *,SEARCH=(PRD2.CONFIG,PRD2.SCEEBASE,PRD1.BASE)
 // EXEC DFHCSDUP,SIZE=600K        INIT AND LOAD CICS CSD VSAM FILE
```

Comment

The job above upgrades for CICS/VSE the common CSD file which belongs to the CICS Transaction Server. Be aware that the CICS TS utility DFHCSDUP is used for the upgrade.

```
 DELETE LIST(VSELSTO)
 DELETE LIST(VSELSTOT)
 DELETE GROUP(VSESPO)
 DELETE GROUP(VSESPOT)
 MIGRATE TABLE(DFHPPTCO) TOGROUP(VSESPO)
 MIGRATE TABLE(DFHPCTCO) TOGROUP(VSESPO)
 ADD GROUP(VSESPO) LIST(VSELSTO)
 COPY GROUP(DFHTERM) TO(VSESPOT)
 DELETE TERMINAL(CBRF) GROUP(VSESPOT)
 COPY GROUP(VSETERM) TO(VSESPOT)
 DELETE TERMINAL(CBRF) GROUP(VSESPOT)
 APPEND LIST(DFHLIST) TO(VSELSTOT)
 REMOVE GROUP(DFHTERM) LIST(VSELSTOT)
 ADD GROUP(VSESPOT LIST(VSELSTOT)
 APPEND LIST(VSELSTOT) TO(VSELSTO)
 ADD GROUP(DFHCOMP2) LIST(VSELSTO)
 ADD GROUP(VSETYPE) LIST(VSELSTO)
 ADD GROUP(VSETERM) LIST(VSELSTO)
 ADD GROUP(VSETERM1) LIST(VSELSTO)
 ADD GROUP(CEE) LIST(VSELSTO)
 ADD GROUP(TCPIP) LIST(VSELSTO)
 LIST ALL
/*
/&
$$ EOJ
```

Comment

DUMPEXT ensures that the CICS Dump Analysis Routine (DFHDAP) is known to the system.

Figure 59. Skeleton SKPREPSO (Preparation Jobs for CICS/VSE 2.3) (Part 6 of 7)

```
$$ JOB JNM=DUMPEXT,CLASS=0,DISP=D
// JOB DUMPEXT EXTEND EXTERNAL ROUTINE FILE
// LIBDEF *,SEARCH=(PRD2.CICSOLDP,PRD1.BASE)
// UPSI 1
// SETPARM DASD=''                 INITIALIZE DASD PARAMETER
// SETPARM XNCPU=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC PROC=CPUVAR&XNCPU,DASD      GET DASD VALUE
// EXEC PROC=DTRINFOA            GET INFO ANALYSIS ASSGNS

* ------------------------------------------------------------
* YOU WILL GET MESSAGE
* 4433D EQUAL FILE ID IN VTOC  BLNXTRN  SYS017=XXX  SYSWK1
* INFO.ANALYSIS.EXT.RTNS.FILE
* ENTER 'DELETE'
* ------------------------------------------------------------
// IF DASD EQ GFBA THEN
// GOTO INFBA
/. INCKD                INITIALIZE EXTERNAL ROUTINES FILE ON CKD
// EXEC DITTO
$$DITTO CSQ BLKFACTOR=1,FILEOUT=BLNXTRN
ANEXIT DFHPD410 CICS DUMP ANALYZER
ANEXIT DFHDAP   CICS 2.3 DUMP ANALYZER
ANEXIT IJBXCSMG ANALYSE CONSOLE BUFFER
ANEXIT IJBXDBUG ANALYSE STANDALONE DUMP ROUTINE
ANEXIT IJBXSDA  SDAID BUFFER FORMATTING ROUTINE
/*
$$DITTO EOJ
/*
// GOTO EXIT
/. INFBA                INITIALIZE EXTERNAL ROUTINES FILE ON FBA
// EXEC DITTO
$$DITTO CSQ BLKFACTOR=1,FILEOUT=BLNXTRN,CISIZE=512
ANEXIT DFHPD410 CICS DUMP ANALYZER
ANEXIT DFHDAP   CICS 2.3 DUMP ANALYZER
ANEXIT IJBXCSMG ANALYSE CONSOLE BUFFER
ANEXIT IJBXDBUG ANALYSE STANDALONE DUMP ROUTINE
ANEXIT IJBXSDA  SDAID BUFFER FORMATTING ROUTINE
/*
$$DITTO EOJ
/*
/. EXIT                END OF INFO/ ANALYSIS FILE INITIALIZE
/&
$$ EOJ
```

Figure 59. Skeleton SKPREPSO (Preparation Jobs for CICS/VSE 2.3) (Part 7 of 7)

Skeleton SKJOURO

The skeleton is used to format the system and user journal datasets that are used with the CICS/VSE 2.3 system. This job reserves disk space for the system and journal datasets on the DOSRES volume. You must add standard labels to the procedure DTRCICSO, which are defined in the member SKPREPCO.

To activate CICS journaling, you must:

1. Submit the job SKJOURO to format the journal datasets.
2. Submit the member DFHJCTCO to define the journal control table.
3. Modify the member DFHSITCO so that `JCT=CO`.
4. Include this assignment in your CICS startup job CICSOLD (contained in skeleton SKCICSOL):
   ```
   // ASSGN SYS029,DISK,VOL=DOSRES,SHR
   ```
5. Shut down and then restart your CICS/VSE 2.3 system.

```
* $$ JOB JNM=CICSJOUR,CLASS=A,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB CICSJOUR    FORMAT CICS 2.3 JOURNAL DATA SETS
// SETPARM DASD=''
// SETPARM XNCPU=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC PROC=CPUVAR&XNCPU,DASD
// IF DASD=3380 THEN
// GOTO D3380
// IF DASD=3390 THEN
// GOTO D3390
// IF DASD=GFBA THEN
// GOTO DGFBA
/. D3380
* -------------------------------------------------------------------*
* FORMAT SYSTEM JOURNAL: DFHJ01A                                     *
* -------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.SYSTEM.LOG.A',0,SD
// EXTENT SYS029,DOSRES,1,0,26055,60
// ASSGN SYS029,DISK,VOL=DOSRES,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
* -------------------------------------------------------------------*
* FORMAT SYSTEM JOURNAL: DFHJ01B                                     *
* -------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.SYSTEM.LOG.B',0,SD
// EXTENT SYS029,DOSRES,1,0,26115,60
// ASSGN SYS029,DISK,VOL=DOSRES,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
* -------------------------------------------------------------------*
* FORMAT USER   JOURNAL: DFHJ02A                                     *
* -------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.USER.JOURNAL.A',0,SD
// EXTENT SYS029,DOSRES,1,0,26175,60
// ASSGN SYS029,DISK,VOL=DOSRES,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
```

Figure 60. Skeleton SKJOURO (Format CICS/VSE 2.3 System and User Journal Datasets) (Part 1 of 3)

```
* ------------------------------------------------------------------*
* FORMAT USER   JOURNAL: DFHJ02B                                     *
* ------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.USER.JOURNAL.B',0,SD
// EXTENT SYS029,DOSRES,1,0,26235,60
// ASSGN SYS029,DISK,VOL=DOSRES,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
// GOTO EOJ
/. D3390
* ------------------------------------------------------------------*
* FORMAT SYSTEM JOURNAL: DFHJ01A                                     *
* ------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.SYSTEM.LOG.A',0,SD
// EXTENT SYS029,DOSRES,1,0,16440,60
// ASSGN SYS029,DISK,VOL=DOSRES,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
* ------------------------------------------------------------------*
* FORMAT SYSTEM JOURNAL: DFHJ01B                                     *
* ------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.SYSTEM.LOG.B',0,SD
// EXTENT SYS029,DOSRES,1,0,16500,60
// ASSGN SYS029,DISK,VOL=DOSRES,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
* ------------------------------------------------------------------*
* FORMAT USER   JOURNAL: DFHJ02A                                     *
* ------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.USER.JOURNAL.A',0,SD
// EXTENT SYS029,DOSRES,1,0,16560,60
// ASSGN SYS029,DISK,VOL=DOSRES,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
* ------------------------------------------------------------------*
* FORMAT USER   JOURNAL: DFHJ02B                                     *
* ------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.USER.JOURNAL.B',0,SD
// EXTENT SYS029,DOSRES,1,0,16620,60
// ASSGN SYS029,DISK,VOL=DOSRES,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
// GOTO EOJ
/. DGFBA
* ------------------------------------------------------------------*
* FORMAT SYSTEM JOURNAL: DFHJ01A                                     *
* ------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.SYSTEM.LOG.A',0,SD
// EXTENT SYS029,SYSWK1,1,0,970752,3072
// ASSGN SYS029,DISK,VOL=SYSWK1,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
* ------------------------------------------------------------------*
* FORMAT SYSTEM JOURNAL: DFHJ01B                                     *
* ------------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.SYSTEM.LOG.B',0,SD
// EXTENT SYS029,SYSWK1,1,0,973824,3072
// ASSGN SYS029,DISK,VOL=SYSWK1,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
```

Figure 60. Skeleton SKJOURO (Format CICS/VSE 2.3 System and User Journal Datasets) (Part 2 of 3)

```
* -----------------------------------------------------------------*
* FORMAT USER   JOURNAL: DFHJ02A                                   *
* -----------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.USER.JOURNAL.A',0,SD
// EXTENT SYS029,SYSWK1,1,0,976896,3072
// ASSGN SYS029,DISK,VOL=SYSWK1,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
* -----------------------------------------------------------------*
* FORMAT USER   JOURNAL: DFHJ02B                                   *
* -----------------------------------------------------------------*
// DLBL JOURNAL,'CICSO.USER.JOURNAL.B',0,SD
// EXTENT SYS029,SYSWK1,1,0,979968,3072
// ASSGN SYS029,DISK,VOL=SYSWK1,SHR
// EXEC DFHJCJFP,SIZE=AUTO
/*
// GOTO EOJ
/. EOJ
/&
* $$ EOJ
```

Figure 60. Skeleton SKJOURO (Format CICS/VSE 2.3 System and User Journal Datasets) (Part 3 of 3)

Chapter 10. Maintaining VTAM Application Names and Startup Options

Maintaining VTAM Application Names

The dialog *Maintain VTAM Application Names* helps you maintain VTAM application names: To access the dialog, start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
5 (Maintain VTAM Application Names)

Administrator Fast Path: 25	Synonym Default: ____ Yours:

You get the panel *VTAM APPLID Maintenance: APPLID List* as shown in Figure 61.

```
COM$APPA          VTAM APPLID MAINTENANCE: APPLID LIST



OPTIONS:     2 = ALTER AN APPLID            5 = DELETE AN APPLID


   OPT       APPLID   APPLICATION  APPLICATION   DEFAULT
                         TYPE       PROPERTY     LOGAPPL
    _       DBDCCICS     CICS       LOGAPPL       X
    _       PRODCICS     CICS       LOGAPPL       _
    _       OLDCICS      CICS       LOGAPPL       _
    _       POWER        RJE                      _
    _       PNET         PNET                     _
    _       ________                              _
    _       ________                              _
    _       ________                              _
    _       ________                              _
    _       ________                              _
    _       ________                              _

PF1=HELP       2=REDISPLAY  3=END                   5=PROCESS    6=ADD APPL
```

Figure 61. Panel for VTAM APPLID Maintenance

The panel lists the application names (APPLIDs) of the VTAM applications installed on your system. The APPLIDs listed are defined as minor nodes of the VTAM application major node VTMAPPL. The applications shown are of type:

- CICS

  for primary CICS Transaction Server (DBDCCICS) or additional CICS systems (PRODCICS or OLDCICS, for example).
- RJE

  for the standard VSE/POWER RJE (Remote Job Entry) definition.
- PNET

  for the VSE/POWER networking support program.

Other *possible* application types are:

- TCP/IP

  if you are using the z/VSE base program TCP/IP for VSE/ESA.
- PSF

  if you are using the z/VSE optional program PSF/VSE (Print Services Facility/VSE).
- SELF-DEFINED

  for user-defined applications.

If you define your own applications (SELF-DEFINED), you have to include the application macro definition in a special library member in VSE/ICCF library 2. This member is named E$$VTMAP and included in the VTAM major node VTMAPPL during generation. If you have that member in your private VSE/ICCF library, it is retrieved from there and not from library 2. “Application Major Node” in the manual *z/VSE Networking Support* provides further details about this facility.

Application property LOGAPPL, as shown for application type CICS, indicates that you can set up a direct sign on to that application. An 'X' in the last column indicates that this APPLID is taken as default value for the LOGAPPL parameter for terminal configuration. The dialog offers the following functions:

- Add (**PF6**)
- Alter (Option **2**)
- Delete (Option **5**)

After entering your changes, press **PF5**. You get the *Job Disposition* panel to submit the job created to batch, or file it in your VSE/ICCF primary library, or both.

Maintaining VTAM Startup Options

With the dialog *Maintain VTAM Startup Options* you maintain VTAM parameters for startup. To access the dialog, start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
6 (Maintain VTAM Startup Options)

```
Administrator
Fast Path:   26
```

```
Synonym
Default:    ____    Yours:
```

You get the panel *VTAM Start Options Maintenance* that allows you to maintain the following parameters stored in the VTAM startup member ATCSTR00:

HOSTSA

This is the SNA subarea number of this host. The value (in hexadecimal) you can specify depends on the naming convention selected during initial installation. The 2-digit subarea naming convention allows a value from 1 to 255; the 4-digit subarea naming convention from 1 to 65535. Note, however, that for a value greater 511 each subarea range of 256 requires an additional 7KB of VTAM buffer.

z/VSE supports per default 511 subareas. This is the default value in the start option MXSUBNUM.

If you change the HOSTSA value, the system renames **on request** all VTAM resource names. The suffix of the resource names is changed to the HOSTSA value.

PROMPT

Enter 1 if you want the operator to be prompted for entering the startup options during system startup.

Enter 2 if you want the system to take the values as defined on the panel. This results in an automatic startup of VTAM. No operator intervention is required. z/VSE creates VTAM startup member ATCSTR00 and stores it in VSE/ICCF library 51. You can apply permanent changes to the VTAM startup book(s) (if they were created via dialog) by using member E$$VTMST in VSE/ICCF library 2. Member E$$VTMST is automatically included in startup member ATCSTR00. "Values Entered through a Dialog" in the manual *z/VSE Networking Support* provides further details.

NETID

This is the 1 to 8 character name of the network this VTAM is part of. The name should be unique within interconnected networks.

After typing in your changes, press **ENTER**. You get the *Job Disposition* panel to submit the job created to batch, or file it in your VSE/ICCF primary library, or both.

Chapter 11. Managing VSE/VSAM Files and Catalogs

Related Section:

- "Using FlashCopy to Backup VSE/VSAM Datasets" on page 292

Overview of File and Catalog Management Dialogs

For the model system administrator (SYSA), the Interactive Interface offers the *File and Catalog Management* dialog. The user profile of SYSA allows the definition, deletion, and processing of VSE/VSAM files and user catalogs.

For the model programmer (PROG), the Interactive Interface offers the *File Management* dialog. The user profile of PROG allows the definition, deletion, and processing of VSE/VSAM files, **but not** of user catalogs.

Some dialogs process the information immediately. Others create a job. You can submit the job for processing or store it as a VSE/ICCF library member in your default VSE/ICCF primary library. Column positions of some parameters in these jobs are essential. Be careful not to change any parameter positions when looking at a job stored as a VSE/ICCF library member.

To access the dialog, start with the *z/VSE Function Selection* panel. As administrator (SYSA), select:

2 (Resource Definition)
2 (File and Catalog Management)

```
Administrator
Fast Path:   22
```

```
Synonym
Default:     vsam   Yours:
```

The programmer (PROG) must choose selection **6** of the *z/VSE Function Selection* panel. The default synonym is the same as for the administrator. Below, the selections for the administrator are shown. For the default programmer (PROG), only the first four selections are displayed.

The panel displayed for the administrator offers six selections:

1 (Display or Process a File)
2 (Define a New File)
3 (Define a Library)
4 (Define an Alternate Index or Name)
5 (Display or Process a Catalog, Space)
6 (Define a New User Catalog)

Note: For selections 1 through 4, a default catalog name is displayed as defined in the user profile. This name can be changed on the panel.

The dialog authorization for the administrator and programmer is based on a general authorization concept. You can have the authority to:

- Define/delete files.
- Process catalogs.

This authorization is part of the user profile. When you define a user profile, you specify whether the user has the authority to define/delete files and the authority to process catalogs. This does not depend on whether the user is an administrator or programmer.

Table 5 illustrates the selections which the panel displays and which you can access based on the authorization you have. If the panel displays a selection which you cannot access and you enter that selection number, the dialog displays an error message.

Table 5. Relationship Between VSE/VSAM Authorization in User Profile and Dialog Selections

Define/Delete Files	Process Catalogs	Selections Displayed and Accessible
YES	YES	All selections displayed. All selections can be accessed.
YES	NO	Selections 1 - 4 displayed. Selections 1 - 4 can be accessed.
NO	YES	Selections 1 - 6 displayed. Only selections 1, 5, and 6 can be accessed.
NO	NO	Selections 1 - 4 displayed. Only selection 1 can be accessed.

Displaying or Processing a File

The dialog *Display or Process a File* provides a FULIST that shows the file IDs and file names of all files in the specified catalog. Use **PF7** and **PF8** to scroll through the list. Use **PF9** to list a subset of the files by entering a prefix. With **PF2** you can refresh the panel display. To locate a particular file, enter the file ID in the LOCATE FILE ID field. The FILE TYPE field contains either the letter *A* or *B* to show the type of file.

```
A  -  Alternate index
B  -  Base file
```

The options you can choose are at the top of the FULIST. Enter an option number in the OPT column to the left of the file ID you want to process. The options available are listed below. They are described in detail in the manual *VSE/ESA Programming and Workstation Guide* under “Display or Process a File”.

1 (Show) Displays details about the characteristics of a VSE/VSAM file or alternate index.

2 (Sort) Sorts a VSE/VSAM file. You must have:
- The z/VSE optional program DFSORT/VSE or a compatible program installed.
- Both input and output files already defined in the catalog.

3 (Print) Prints one or more records of a VSE/VSAM file on the system printer.

4 (Copy) Copies all or part of a file to another file. You can also copy a VSE/VSAM file to and from tape.

5 (Delete) Deletes a VSE/VSAM file or an alternate index and name. You **cannot** delete system files.

6 (Verify) Compares the end-of-file information in the catalog with the end-of-file indicator(s) in the file. If the information does not agree, the catalog information is corrected. You **cannot** verify an alternate index.

7 (Load)
Loads data from a VSE/ICCF library member into a base file, or loads an alternate index from a base file.

Defining a New File

With the dialog *Define a New File* you can create a new VSE/VSAM file in the catalog specified. To access the dialog, start with the *z/VSE Function Selection* panel and select:

Administrator	Synonym		
Fast Path: 222	Default:	____	Yours:

If you want to specify a catalog and not use the default catalog, select fast path **22**.

The dialog displays several panels. You need to enter different file characteristics, depending on the type of file you are defining.

The dialog defines the new file and adds a label to the system standard label area with the file name, file ID, and catalog name of the new file. It also adds label information to the VSE/VSAM label procedure STDLABUP in IJSYSRS.SYSLIB.

You need the following information:

FILE ID
Enter up to five segments for the file ID. You cannot enter more than 38 characters, including dots.

FILE NAME
Enter **1 - 7** alphameric characters. The first character must be alphabetic.

FILE ORGANIZATION
1 - Non-keyed (ESDS)
2 - Keyed (KSDS)
3 - Numbered (RRDS)
4 - Numbered (VRDS)
5 - Sequential (SAM ESDS)

FILE ADDRESSABILITY
1 - No extended addressing. This is the default.
2 - Extended addressing. Define a VSE/VSAM KSDS file larger than 4 GB.

FILE ACCESS
For the VSE/VSAM Share option, specify:
1 - Multiple Read OR Single Write
2 - Multiple Read AND Single Write
3 - Multiple Read AND Write (no integrity)
4 - Multiple Read AND Write (with integrity)

FILE USAGE
1 - Data file (NOREUSE)
2 - Work file (REUSE)

If the catalog owns space on more than one volume, at this point a list is displayed which shows these volumes and their device type. You can then select the volume you want the primary space allocated on and the volume(s) you want the secondary space allocated on.

EXPIRATION DATE
Enter four digits for the year, and three digits for the day of the year (**YYYYDDD**).

ALLOCATION UNIT
Required only for **CKD disk devices**. Specify one of the following:
1 - Cylinder
2 - Track

For FBA devices, the allocation unit "Block" is used automatically.

PRIMARY and SECONDARY ALLOCATION
The number of allocation units for the initial (primary) and subsequent (secondary) allocations.

CONTROL INTERVAL SIZE
Specify the Control Interval size of the data component for all file types. For file types that have indexes, the value of the index component is calculated by VSAM.

AVERAGE and MAXIMUM RECORD SIZE
The average and maximum length of the data record, in bytes. For RRDS files, the average and maximum record sizes are the same. If you are defining a **sequential** file, you do **not** need this information.

DATA COMPRESSION
Data compression is available for VSE/VSAM files of type ESDS, KSDS, and VRDS. By specifying **1**, data compression will be enabled.

The manual *z/VSE Planning* under "Data Compression Support" provides introductory information about the data compression support.

Additional information required for specific file types:

Keyed (KSDS) Files

For keyed (KSDS) files, specify KEY LENGTH and POSITION.

Enter the key length from **1 - 255**. The key position is the offset of the key from the beginning of the record.

Sequential Files

If you define a sequential file, specify the following file characteristics:

RECORD FORMAT
- **1** - Fixed, unblocked
- **2** - Fixed, blocked
- **3** - Variable, unblocked
- **4** - Variable, blocked
- **5** - Undefined
- **6** - No control interval format

RECORD SIZE
Fixed record formats only (RECORD FORMAT options 1 and 2). Enter the record length.

BLOCK SIZE
Fixed, blocked format only (RECORD FORMAT option 2). Enter the block length.

AVERAGE RECORD SIZE
Variable length and undefined formats only (RECORD FORMAT options 3, 4, or 5). Enter the average length of the record.

MAXIMUM RECORD SIZE
Variable length and undefined formats only (RECORD FORMAT options 3, 4, or 5). Enter the maximum length of the record.

On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements as a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Note: If you use DTSECTAB to protect STDLABUP, your user ID **must** be included in DTSECTAB. If it is not, the job fails when it tries to update label information.

Defining a Library

With the dialog *Define a Library* you can create a VSE library in VSE/VSAM managed space. To access the dialog, start with the *z/VSE Function Selection* panel and select:

Administrator Fast Path: 223	Synonym Default: ____ Yours:

If you want to specify a catalog and not use the default catalog, select fast path 22.

You need the following information:

LIBRARY NAME
: Specify the library file name. 'VSE.file name.LIBRARY' is the default for the file ID created by the dialog.

PRIMARY ALLOCATION
: Enter the number of 1K library blocks.

SECONDARY ALLOCATION
: Enter the number of 1K library blocks.

EXTENTS
: Enter either **1** (for a maximum of 16 extents) or **2** (for a maximum of 32 extents). You can specify a maximum of 32 extents when the library is a multi-volume file and is defined in space managed by VSE/VSAM. If you specify a maximum of 32 extents (MAX32), you must also select SECONDARY ALLOCATION on the next *Select Space* panel.

If the catalog owns space on more than one volume, a list is displayed that shows these volumes and their device type code. You can then select the volume you want the primary space allocated and the volume(s) you want the secondary space allocated.

On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements in a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Notes:

1. If you use DTSECTAB to protect STDLABUP, your user ID **must** be in DTSECTAB. If it is not, the job fails when it updates label information.
2. To access a library you need at least one sublibrary. Use the librarian (LIBR) program to define sublibraries. "Using VSE Libraries" in the manual *z/VSE Guide to System Functions* describes program LIBR in detail.

Defining an Alternate Index or Name

With the dialog *Define an Alternate Index or Name* you can create either an alternate index or an alternate name for an existing VSE/VSAM file. A FULIST displays the file IDs and names of the files in the catalog. To access the dialog, start with the *z/VSE Function Selection* panel and select:

Administrator	Synonym
Fast Path: 224	Default: ____ Yours:

If you want to specify a catalog and not use the default catalog, select fast path 22. Use **PF7** and **PF8** to scroll through the list. Use **PF9** to list a subset of the files by entering a prefix. With **PF2** you can refresh the display. To locate a particular file, enter the file ID in the LOCATE FILE ID field.

Alternate Index

This task defines an alternate index over an existing base file. When you define an alternate index, two things are defined:

1. The alternate index cluster.
2. The path.

The name and ID you specify become the *path* name and ID. The system generates the name of the alternate index cluster internally. You need the following information:

ALTERNATE INDEX ID and NAME
Specify the ID and the name of the alternate index.

KEY POSITION and LENGTH
Specify the position and length of the alternate key within the base record. The key length can be **1 - 255**.

KEYS Specify the maximum number of non-unique keys in the alternate index. The dialog uses this value to calculate the maximum record length of the alternate index file.

The dialog adds a label to the system standard label area with the file name, file ID, and catalog name of the alternate index. It also adds label information to the VSE/VSAM label procedure STDLABUP.

On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements as a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Note: If you use DTSECTAB to protect STDLABUP, your user ID **must** be in DTSECTAB. If it is not, the job fails when it updates label information.

Alternate Name

This task defines an alternate name for the file. It also adds label information to the VSE/VSAM label procedure STDLABUP.

If a file does not have a file name (it has no label in the system standard label area), you can use this task to define the file name. You **should not** define alternate names for libraries.

You only need to specify the alternate file name.

On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements in a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Note: If you use DTSECTAB to protect STDLABUP, your user ID **must** be in DTSECTAB. If it is not, the job fails when it updates label information.

Displaying or Processing a Catalog or Space

The dialog *Display or Process a Catalog, Space* provides a FULIST that shows the catalog IDs and names in the system. To access the dialog, start with the *z/VSE Function Selection* panel and select:

```
Administrator          Synonym
Fast Path:  225        Default:    ____    Yours:
```

Use **PF7** and **PF8** to scroll through the list. With **PF2** you can refresh the display.

You can select the following options:
1 (Show space)
2 (Define alternate name)
3 (Print catalog contents)
4 (Define space)
5 (Delete catalog)
6 (Delete space)

Note: The following applies for options 2 and 5: If you use DTSECTAB to protect STDLABUP, your user ID **must** be in DTSECTAB. If it is not, the job fails when it updates label information.

Show Space

This task displays details about the space owned by the catalog selected. A panel lists the volumes owned by the catalog. It displays the allocated, used, and free space on each volume.

Define Alternate Name

This task defines an alternate name for the catalog. The dialog defines the name and adds a label to the system standard label area with the alternate name. It also adds label information to the VSE/VSAM label procedure STDLABUP.

You should use alternate names for catalogs carefully. On the *File and Catalog Management* panel, if you:

- Select options 1 or 4

  AND
- Specify an alternate catalog name in the CATALOG NAME field

The FULIST only displays file names for files which are defined with the alternate catalog name. The FULIST displays *NONE* as the file name for files defined in the same catalog with a different catalog name.

You only need to specify the alternate catalog name.

On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements in a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Print Catalog Contents

This task creates a LISTCAT of the selected catalog. You do not have to specify any information. On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is submitted

Option **1** stores the job control statements as a VSE/ICCF library member in your default primary library.

Option **2** submits the job automatically. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Define Space

This task defines VSE/VSAM space that is to be used for the allocation of files. It is recommended that you define space owned by a catalog on the same volume on which the catalog resides. You need the following information:

VOLUME NAME
Enter the six character VOLID of the disk where the space should be defined.

ALL FREE SPACE
Specify whether you want all available space on the volume for VSE/VSAM:
1 - YES
2 - NO

If you specify 1 (YES), all free space (up to 16 extents) on the volume is dedicated to VSE/VSAM.

If you specify 2 (NO), a panel displays the free extents on the volume. Select one extent. Enter the beginning allocation and the amount of space to be allocated.

In some circumstances, VSE/VSAM rounds the specified values to a higher number. If the rounded extent exceeds the original one, the space definition fails. To avoid this, choose values which result in a smaller extent than the one shown.

SPACE AVAILABLE TO CURRENT FILES
Specify whether the files currently owned by the catalog can access the new space for secondary allocation:
1 - YES
2 - NO

If you specify 1 (YES), the dialog alters the catalog entries of the current files so new space can be used for secondary allocation. The dialog changes the catalog entries, if:

- The secondary allocation for the file is greater than 0.
- You define new space on disk devices with the same device type code as the primary allocation for the file(s).

If a current file already accesses space on a volume, it keeps that access when you define new space on the same volume.

If you specify 2 (NO), current files **cannot** access the new space.

On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements in a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Notes:

1. For CKD disk devices, the units of allocation are *cylinders* and **not** tracks.
2. If you are using emulated FBA disks or virtual FBA disks, VSE/VSAM can use as VSE/VSAM space the first 4194240 blocks (2 GB) in units of 960 blocks.
3. If you are using FBA-SCSI disks, VSE/VSAM can use as VSE/VSAM space the first 33546240 blocks (16 GB) in units of 30720 blocks. For further details, refer to the manual *z/VSE Planning*, SC33-8221.

Delete Catalog

This task deletes a VSE/VSAM catalog or alternate catalog name. The dialog removes the user catalog entry from the VSE/VSAM master catalog. Before you delete a catalog, you should do the following:

- Delete all files which the catalog owns.
- If the catalog owns space on more than one volume, first delete the space on the volumes other than the catalog volume.

The dialog deletes the catalog and removes the label from the system standard label area and the VSE/VSAM label procedure STDLABUP. If the catalog has alternate names, specify what you want to delete:

1 - Delete catalog name only
2 - Delete actual catalog, including alternate names

Verify that the catalog is the one you want to delete.

On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements in a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Delete Space

This task deletes VSE/VSAM data spaces. A panel displays a list of the volumes owned by the catalog. It shows the allocated, used, and free space on each volume.

Enter **5** in the OPT column next to the volume you want to select. On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements in a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Defining a New User Catalog

With the dialog *Define a New User Catalog* you can create a new user catalog and, optionally, space for file allocation.

Data Compression Support: The dialog automatically creates for each user catalog a CCDS (Compression Control Data Set) which is named VSAM.COMPRESS.CONTROL. This cluster is required to enable data compression for ESDS, KSDS, and VRDS files defined in the catalog. Under “Data Compression Support”, the manual *z/VSE Planning*provides an overview about the data compression support available.

To access the dialog, start with the *z/VSE Function Selection* panel and select:

```
Administrator              Synonym
Fast Path:  226            Default:    ____    Yours:
```

The dialog adds a label to the system standard label area with the catalog name. It also adds label information to the VSE/VSAM label procedure STDLABUP.

Space which belongs to a catalog should be on the same volume as the catalog itself. When you define the catalog, you can use only part of a volume and then use more of the volume later. However, it is better if you take as much space on the volume as you will need. Additional space may not be available later. You need the following information:

USER CATALOG ID and NAME
For the new catalog, enter up to five segments for the file ID. You cannot enter more than 38 characters, including dots. For the name, enter **1 - 7** alphameric characters.

Note: The catalog name IJSYSUC is reserved for system use.

VOLUME NAME
Enter the six character volume ID of the disk where the catalog will be defined.

ALL FREE SPACE
Specify whether you want all available space on the volume for VSE/VSAM:
1 - YES
2 - NO

If you specify 1 (YES), the dialog uses all free space (up to 16 extents) on the volume for both the catalog and files. VSE/VSAM determines the size of the space reserved for the catalog and for the files.

If you specify 2 (NO), you are defining space for the catalog only, not for file allocation. A panel displays the free extents on the volume. Select one extent. Enter the beginning allocation and the amount of space to be allocated.

Note: Make sure that you do not allocate too much space for the catalog, so that there is not enough space for the VSE/VSAM data.

In some circumstances, VSE/VSAM rounds the specified values to a higher number. If the rounded extent exceeds the original one, the space definition fails. To avoid this, choose values which result in a smaller extent than the one shown.

If you specify 2 (NO), you must define space for the files explicitly. Use the *Display or Process a Catalog, Space* dialog for that purpose. Refer to "Displaying or Processing a Catalog or Space" on page 248 for details.

On the *Job Execution* panel, select:

1 - Delayed, Submission is handled by user
2 - Immediate, Job is executed

Option **1** stores the job control statements in a VSE/ICCF library member in your default primary library. The default member name is **F$xxxx**, where xxxx is your user ID. You can change the name on the panel.

Option **2** automatically runs the job online. The terminal is locked until the job finishes. It also stores the VSE/ICCF member **F$xxxx.P** (xxxx is your user ID) in your default primary library. If there are no errors, the member contains one record with an asterisk (*). If errors occurred, the control statements and VSE/VSAM (Access Method Services) error messages are stored in this member. You are notified if you should review the contents of the library member.

Note: If you use DTSECTAB to protect STDLABUP, your user ID **must** be in DTSECTAB. If it is not, the job fails when it updates label information.

Chapter 12. Managing Non-VSE/VSAM Libraries and User File Labels

To manage libraries in **non-VSE/VSAM** space, z/VSE provides the following skeletons:
- SKLIBDEF (for defining libraries)
- SKLIBEXT (for extending libraries)
- SKLIBDEL (for deleting libraries)

To create standard labels for non-VSE/VSAM user files, z/VSE provides skeleton STDLABUS.

These skeletons are described on the following pages.

To manage libraries in **VSE/VSAM** space, z/VSE provides dialogs as described in Chapter 11, "Managing VSE/VSAM Files and Catalogs," on page 241.

Defining a VSE User Library in Non-VSE/VSAM Space

The **SKLIBDEF** skeleton defines a VSE user library in non-VSE/VSAM space. The skeleton has two major steps:
1. Add standard label for the new library.
2. Create the library.

The skeleton is shipped in library 59. First copy it to your primary library and edit the copied file. Refer to "Copying Skeletons" on page 6 for information on copying skeletons.

Figure 62 on page 256 shows the skeleton. The variables you should change are highlighted in color. They are described below the figure to help you make the correct changes.

After you make the changes, update the standard label skeleton STDLABUS. Add the label for the library which SKLIBDEF defines. Refer to "Creating Standard Labels for Non-VSE/VSAM User Files" on page 260 for information about STDLABUS.

Note: To access a library you need at least one sublibrary. Use the librarian (LIBR) program to define sublibraries. "Using VSE Libraries" in the manual *z/VSE Guide to System Functions* describes the librarian program LIBR in detail.


```
* $$ JOB JNM=DEFLIBR,CLASS=0,DISP=D
// JOB DEFLIBR DEFINE LIBRARY IN NON VSAM SPACE
// OPTION STDLABEL=ADD
// DLBL --V001--,'--V002--',99/366,SD
// EXTENT ,--V101--,1,--V102--,--V103--,--V104--
// EXTENT ,--V101--,2,--V102--,--V103--,--V104--
/*
// EXEC LIBR
DEFINE LIB=--V001-- R=Y
/*
/&
* $$ EOJ
END OF MEMBER
```

Figure 62. Define VSE User Library in Non-VSE/VSAM Space (SKLIBDEF Skeleton)

Change the **--Vxxx--** variables in the DLBL, EXTENT, and DEFINE LIB statements.

--V001--
: This is the file name of the library. Specify **1 - 7** alphanumeric characters.

  Note: Change --V001-- in both the DLBL and DEFINE LIB statements.

--V002--
: This is the file identification. Specify **1 - 44** alphanumeric characters. Do **not** delete the single quotes (′ ′) around the variable.

--V101--
: The volume number where the library will reside. It must be **6** characters. The value can be the same or different for each extent.

--V102--
: This is the extent sequence number (**0 - 15**). You can have up to sixteen extents defined on several disks. The disks must all be of the same type.

--V103--
: This is the start location of the library in tracks or blocks. The value can be different for each extent.

--V104--
: This is the amount of space to be allocated for the library on the first volume. The value can be different for each extent. Specify the value in tracks or blocks.

After you have modified the skeleton, enter the following command from the editor's command line:

```
@DTRSEXIT
```

This command calls a macro that deletes specific comments from the skeleton. You should do this *before* filing the skeleton.

Extending a VSE User Library in Non-VSE/VSAM Space

The **SKLIBEXT** skeleton extends a VSE user library in non-VSE/VSAM space. It is recommended that you use a MINI startup for extending a library. This is to ensure that the library is not in an active LIBDEF chain.

The skeleton has five major steps:

1. Back up the library to tape.
2. Delete the library.
3. Delete standard label for the library.
4. Add new standard label for the extended library.
5. Create library and restore the library from tape.

The skeleton is shipped in VSE/ICCF library 59. First copy it to your &iccf primary library and edit the copied skeleton. Refer to “Copying Skeletons” on page 6 for information on copying skeletons.

Figure 63 shows the skeleton. Each section of the skeleton is shown in separate parts of the figure. The variables you should change are highlighted in color. A description of the changes follows each part of the figure.

After you make the changes, update the standard label skeleton STDLABUS. Change the label for the library which SKLIBEXT updates. Refer to “Creating Standard Labels for Non-VSE/VSAM User Files” on page 260 for information about STDLABUS.

```
* $$ JOB JNM=EXTLIBR,CLASS=0,DISP=D
// JOB EXTLIBR EXTEND LIBRARY IN NON VSAM SPACE
*    THIS FUNCTION USES A TAPE FOR OUTPUT
*    MOUNT TAPE REEL --V004-- WITH UNLABELED TAPE ON DEVICE --V003--
*    THEN CONTINUE. IF NOT POSSIBLE CANCEL THIS JOB
*    WARNING: EXISTING TAPE LABEL WILL BE OVERRIDDEN
// PAUSE
// ASSGN SYS005,--V003--
// MTC REW,SYS005
// EXEC LIBR
   BACKUP LIB = --V001--      /* LIBRARY IDENTIFICATION
          RESTORE = ONLINE                     /* RESTORE TYPE
          TAPE = SYS005                        /* TAPEADRESS
/*
// MTC REW,SYS005
// IF $RC > 0 THEN
// GOTO $EOJ
// EXEC LIBR
DELETE LIB=--V001--
/*
// IF $RC > 0 THEN
// GOTO $EOJ
// OPTION STDLABEL=DELETE
--V001--
/*
```

Figure 63. SKLIBEXT Skeleton, Part 1 of 3 (Extend VSE User Library in Non-VSE/VSAM Space)

Change the **--Vxxx--** variables in the comments, the ASSGN, BACKUP LIB, and DELETE LIB statements, and in the statement following the OPTION statement.

--V001--
: This is the file name of the library. Specify **1 - 7** alphanumeric characters.

--V003--
: The tape address (**cuu**) used for the backup.

--V004--
: The volume number of the backup/restore tape. It must be **6** characters.

```
// OPTION STDLABEL=ADD
// DLBL --V001--,'--V002--',99/366,SD

// EXTENT ,--V101--,1,--V102--,--V103--,--V104--
// EXTENT ,--V101--,2,--V102--,--V103--,--V104--
// EXTENT ,--V101--,3,--V102--,--V103--,--V104--
/*
```

Figure 64. SKLIBEXT Skeleton, Part 2 of 3 (Extend VSE User Library in Non-VSE/VSAM Space)

Change the **--Vxxx--** variables in the DLBL and EXTENT statements.

--V001--
: This is the file name of the library. Specify **1 - 7** alphanumeric characters.

--V002--
: This is the file identification. Specify **1 - 44** alphanumeric characters. Do **not** delete the single quotes (′ ′) around the variable.

--V101--
: The volume number where the library will reside. It must be **6** characters. The value can be the same or different for each extent.

--V102--
: This is the extent sequence number (**0 - 15**). You can have up to sixteen extents defined on several disks. The disks must be all of the same type.

--V103--
: This is the start location of the library in tracks or blocks. The value can be different for each extent.

--V104--
: This is the amount of space to be allocated for the library on the first volume. The value can be different for each extent. Specify the value in tracks or blocks.

```
*    THIS FUNCTION USES AN EXISTING TAPE FOR INPUT
*    MOUNT TAPE REEL --V004-- ON DEVICE --V003-- (THE TAPE IS UNLABELED)
*    THEN CONTINUE. IF NOT POSSIBLE CANCEL THIS JOB
*    REPLY 'DELETE' TO MESSAGE '4433D  EQUAL FILE ID IN VTOC ...'

// PAUSE
// ASSGN SYS005,--V003--
// MTC REW,SYS005
// EXEC LIBR
   RESTORE LIB = --V001--
           LIST  = YES
           REPLACE = YES
           TAPE = SYS005
/*
// MTC RUN,SYS005
/&
* $$ EOJ
```

Figure 65. SKLIBEXT Skeleton, Part 3 of 3 (Extend VSE User Library in Non-VSE/VSAM Space)

Change the **--Vxxx--** variables in the comments and the ASSGN, and RESTORE LIB statements.

--V001--
: This is the file name. Specify **1 - 7** alphanumeric characters.

--V003--
: The tape address (**cuu**) used to restore the library.

--V004--
: The volume number of the backup/restore tape. It must be **6** characters.

When you run the job, use the same tape and tape drive you specify in the skeleton.

After you have modified the skeleton, enter the following command from the editor's command line:

```
@DTRSEXIT
```

This command calls a macro that deletes specific comments from the skeleton. You should do this *before* filing the skeleton.

Deleting a VSE User Library in Non-VSE/VSAM Space

The **SKLIBDEL** skeleton deletes a VSE user library in non-VSE/VSAM space. It is recommended that you use a MINI startup for deleting a library. This is to ensure that the library is not in an active LIBDEF chain.

The skeleton has two major steps:
1. Delete the library.
2. Delete the standard label for the library.

The skeleton is shipped in VSE/ICCF library 59. First copy it to your VSE/ICCF primary library and edit the copied file. Refer to "Copying Skeletons" on page 6 for information on copying skeletons.

Figure 66 shows the skeleton. The variable you should change is highlighted in color. It is described below the figure to help you make the correct changes.

After you make the changes, update the standard label skeleton STDLABUS. Delete the label for the library which SKLIBDEL deletes. Refer to "Creating Standard Labels for Non-VSE/VSAM User Files" for information about STDLABUS.

```
* $$ JOB JNM=DELLIBR,CLASS=0,DISP=D
// JOB DELLIBR DELETE LIBRARY IN NON VSAM SPACE
// EXEC LIBR
DELETE LIB=--V001--
/*
// IF $RC > 0 THEN
// GOTO $EOJ
// OPTION STDLABEL=DELETE
--V001--
/*
/&
* $$ EOJ
```

Figure 66. Delete VSE User Library in Non-VSE/VSAM Space (SKLIBDEL Skeleton)

Change the **--V001--** variable in the DELETE LIB statement and the statement following the OPTION statement. --V001-- is the file name. Specify **1 - 7** alphanumeric characters.

After you have modified the skeleton, enter the following command from the editor's command line:

@DTRSEXIT

This command calls a macro that deletes specific comments from the skeleton. You should do this *before* filing the skeleton.

Creating Standard Labels for Non-VSE/VSAM User Files

You use skeleton **STDLABUS** to create standard labels for files you created in space *not managed by VSE/VSAM*. In addition, you can use STDLABUS to add your partition standard labels.

Note: Labels for VSE/VSAM files are *automatically* created or deleted when using the dialogs for file and catalog management. For additional information on z/VSE label processing, consult the manual *z/VSE Planning*. It describes the standard label procedures STDLABEL, STDLABUP, and STDLABUS used by z/VSE under "Standard Label Procedures".

STDLABUS contains VSE/POWER JECL and JCL statements to catalog the procedure in the system library. It is shipped in VSE/ICCF library 59. First copy it to your VSE/ICCF primary library and edit the copied skeleton. Refer to "Copying Skeletons" on page 6 for information on copying skeletons.

The name of the member in the CATALOG statement **must** be STDLABUS because the system standard label procedure STDLABEL calls STDLABUS to load your standard labels.

Figure 67 on page 262 shows the skeleton. Each section of the skeleton is shown in separate parts of the figure. The values you should change are highlighted. The skeleton contains samples for defining labels for the following types of disk files:
- Sequential
- Direct access
- Index sequential

Each sample is enclosed within asterisks (*) and is noted by a heading. DLBL and EXTENT statements follow each sample. Use these statements to define your labels for that particular type of disk file. The DLBL and EXTENT statements contain variables which you replace. The variables are basically the same for each disk file type. They are described below. Differences for specific types of disk files are noted.

The variables you should change in the **DLBL** statements are:

YYYYYYY
This is the file name. Specify 1 - 7 characters.

FFFFFFFF
This is the file-id. Specify 1 - 44 characters.

YYYY/DDD
The retention period of the file where YYYY is the year and DDD is the day.

The variables you should change in the **EXTENT** statements are:

YYYYYY
The SYS number of the file.

VVVVVV
Volume number of the disk where the file resides.

T
File type code. This value differs depending on the type of file. For direct access files, you **must** specify **1**.

For sequential disk files, specify:
1 - Prime data
8 - Prime data with split cylinder (not FBA)

For index sequential disk files, specify:
1 - Prime data
2 - Overflow
4 - Index

S
The file sequence number. This value starts with 0.

BBBBBBBB
Starting track or block for the extent of the file.

NNNNNNNN
Number of tracks or blocks for this extent.

PP
You need this value for sequential disk files only. Specify the split cylinder start track (for split cylinder only).

```
..$$ JOB JNM=CATALOG,DISP=D,CLASS=0
// JOB CATALOG
                   MAKE SURE SYSLST IS ASSIGNED FOR LIBR              C
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG STDLABUS.PROC  DATA=YES     REPLACE=YES
/. STANDARD LABEL SKELETON FOR YOUR STANDARD LABELS
                                                                      C
         SEQUENTIAL DISK FILE SAMPLES.                                C
                                                                      C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
* // DLBL SAMPSD1,'SAMPLE.SEQUENTL.DISK.FILE.ONE',99/366,SD           C
* // EXTENT SYS004,SAMP01,1,0,2400,1200                               C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
    SINGLE EXTENT SD FILE.                                            C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
* // DLBL SAMPSD2,'SAMPLE.SEQUENTL.DISK.FILE.TWO',99/366,SD           C
* // EXTENT SYS004,SAMP01,8,0,2400,1200,6                             C
* // DLBL SAMPSD3,'SAMPLE.SEQUENTL.DISK.FILE.THREE',99/366,SD         C
* // EXTENT SYS004,SAMP01,8,0,2407,1200,14                            C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
    TWO SINGLE EXTENT SD FILES USING SPLIT CYLINDER ON 3390.          C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
* // DLBL SAMPSD4,'SAMPLE.SEQUENTL.DISK.FILE.FOUR',99/366,SD          C
* // EXTENT SYS004,SAMP01,1,0,2400,600                                C
* // EXTENT SYS004,SAMP01,1,1,2400,600                                C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
    MULTI-EXTENT SD FILE.                                             C
                                                                      C
         SEQUENTIAL DISK FILE SKELETON.                               C
                                                                      C
// DLBL YYYYYYY,'FFFFFFFF.FFFFFFFF.FFFFFFFF.FFFFFFFF',YYYY/DDD,SD     C
// EXTENT YYYYYYY,VVVVVV,T,S,BBBBBBBB,NNNNNNNN,PP
```

Figure 67. DTRLABUS Skeleton, Part 1 of 3 (Create Standard Labels)

In the CATALOG statement, the procedure name **must** be STDLABUS. The system standard label procedure STDLABEL calls STDLABUS to load your standard labels.

The three samples are enclosed within asterisks. They are for the following types of sequential disk files:
- Single extent.
- Two single extent using split cylinder on an IBM 3390.
- Multi-extent.

Use the DLBL and EXTENT statements following the three samples to define the labels for your sequential disk files. If you **do not** use these statements, delete them.

```
          DIRECT ACCESS FILE SAMPLE.                                C
                                                                    C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
* // DLBL SAMPDA,'SAMPLE.DIRECT.ACCESS.FILE.A',99/366,DA            C
* // EXTENT SYS002,SAMP01,1,0,100,1200                              C
* // EXTENT SYS002,SAMP01,1,1,1300,1200                             C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
                                                                    C
         DIRECT ACCESS FILE SKELETON.                               C
                                                                    C
// DLBL YYYYYYY,'FFFFFFFF.FFFFFFFF.FFFFFFFF.FFFFFFFF',YYYY/DDD,DA
// EXTENT YYYYYY,VVVVVV,T,S,BBBBBBBB,NNNNNNNN

         INDEX SEQUENTIAL FILE (CREATE) SAMPLE.                     C
                                                                    C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
* // DLBL SAMPISC,'SAMPLE.INDEX.SEQUENTL.FILE.ONE',99/366,ISC        C
* // EXTENT SYS005,SAMP01,4,0,3700,1200                             C
* // EXTENT SYS005,SAMP01,4,1,4900,1200                             C
* // EXTENT SYS005,SAMP01,1,2,6100,1200                             C
* // EXTENT SYS005,SAMP01,1,3,7300,1200                             C
* // EXTENT SYS005,SAMP01,2,4,8500,1200                             C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
                                                                    C
        INDEX SEQUENTIAL FILE (CREATE) SKELETON.                    C
                                                                    C
// DLBL YYYYYYY,'FFFFFFFF.FFFFFFFF.FFFFFFFF.FFFFFFFF',YYYY/DDD,DA
// EXTENT YYYYYY,VVVVVV,T,S,BBBBBBBB,NNNNNNNN

         INDEX SEQUENTIAL (EXTENSION) FILE SAMPLE.                  C
                                                                    C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
* // DLBL SAMPISE,'SAMPLE.INDEX.SEQUENTL.FILE.TWO',99/366,ISE        C
* // EXTENT SYS006,SAMP01,4,0,9700,1200                             C
* // EXTENT SYS006,SAMP01,4,1,10900,1200                            C
* // EXTENT SYS006,SAMP01,1,2,12100,1200                            C
* // EXTENT SYS006,SAMP01,1,3,13300,1200                            C
* // EXTENT SYS006,SAMP01,2,4,14500,1200                            C
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C
                                                                    C
         INDEX SEQUENTIAL (EXTENSION) FILE SKELETON.
                                                                    C
// DLBL YYYYYYY,'FFFFFFFF.FFFFFFFF.FFFFFFFF.FFFFFFFF',YYYY/DDD,DA
// EXTENT YYYYYY,VVVVVV,T,S,BBBBBBBB,NNNNNNNN
```

Figure 68. DTRLABUS Skeleton, Part 2 of 3 (Create Standard Labels)

One sample is enclosed within asterisks for direct access disk files. Use the DLBL and EXTENT statements following the sample to define the labels for your direct access files. If you **do not** use these statements, delete them.

Two samples are enclosed within asterisks for index sequential files (create and extension). Use the DLBL and EXTENT statements following each sample to define the labels for your index sequential (create or extension) files. If you **do not** use these statements, delete them.

```
// OPTION PARSTD
// OPTION PARSTD=F1         FOR STATIC PARTITIONS
// OPTION PARSTD=F2
// OPTION PARSTD=F3
// OPTION PARSTD=F4
// OPTION PARSTD=F5
// OPTION PARSTD=F6
// OPTION PARSTD=F7
// OPTION PARSTD=F8
// OPTION PARSTD=F9
// OPTION PARSTD=FA
// OPTION PARSTD=FB
// OPTION CLASSTD=C         FOR DYNAMIC PARTITIONS
// OPTION CLASSTD=P
// OPTION CLASSTD=Y
// OPTION CLASSTD=Z
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY STDLABUS.PROC     REPLACE=YES
../*
../&
..$$ EOJ
```

Figure 69. DTRLABUS Skeleton, Part 3 of 3 (Create Standard Labels)

After **each** // OPTION statement, add the standard labels for that particular partition. If you do not specify labels for a partition, the area is cleared.

Note: Do not delete any of the // OPTION statements, even if you have no partition standard labels to add. If you do, startup problems are likely to occur.

After making the changes, run the DTRSEXIT macro. This macro deletes specific comments from the skeleton. You should do this before you file the skeleton. On the command line, enter:

```
@DTRSEXIT
```

Chapter 13. Using Virtual Tapes

In z/VSE, a *virtual tape* is a file (or dataset) containing a tape image. You can read from or write to a virtual tape in the same way as if it were a physical tape. A virtual tape can be:

- A VSE/VSAM **ESDS** file on the z/VSE host side.
- A **remote** file on the server side; for example, a Linux, UNIX, or Windows file. To access such a remote virtual tape, a **TCP/IP** connection is required between z/VSE and the remote system.

This chapter describes virtual tapes in these main sections:

What the Virtual Tape Support Consists Of

The Virtual Tape Support consists of three functional components:

- **Virtual Tape Simulator**

  The Virtual Tape Simulator is part of z/VSE and ready for use after z/VSE installation. It controls virtual tape processing independent of where the virtual tape is located (VSE/VSAM or remote). It receives any incoming requests and forwards them to the Virtual Tape Data Handler for processing. It finally provides status information about the function performed.
- **Virtual Tape Data Handler**

  The Virtual Tape Data Handler is part of z/VSE and ready for startup after z/VSE installation. It is required for handling the read or write access to any virtual tape: VSE/VSAM virtual tapes or remote virtual tapes via TCP/IP. When the Virtual Tape Support is activated (VTAPE START command or statement), the Virtual Tape Data Handler is loaded into a dynamic partition, which is the default, or into a static partition (refer also to "Starting and Stopping the Virtual Tape Data Handler" on page 270) . When the Virtual Tape Support is deactivated, the partition is released again.
- **Virtual Tape Server**

  The Virtual Tape Server is required for **remote virtual tapes only**. It is the workstation counterpart to the Virtual Tape Data Handler on the z/VSE side with which it communicates via TCP/IP. The Virtual Tape Server must be installed on a workstation with a Java platform (refer also to "Installing the Virtual Tape Server" on page 266).

Prerequisite For Using the Virtual Tape Support

Before you use the Virtual Tape Support, you should be aware that the z/VSE I/O Supervisor will allocate a 1 MB buffer for *each* virtual tape in the PFIXed System GETVIS area of your z/VSE system. This may cause problems if you run virtual tapes in parallel, and where the real storage available is not greater than 16 MB (under z/VM, for example).

The 1 MB buffer is used to buffer the tape data before it is:
- Written to the virtual tape.
- Read from the virtual tape.

Therefore, before you use the VTAPE START command, you must ensure that there is sufficient PFIXed space available in the System GETVIS area of your z/VSE system.

To check that sufficient PFIXed System GETVIS space is available, you can issue the AR command:

```
MAP SVA
```

If the system has insufficient PFIXed System GETVIS storage, when the VTAPE START command is issued this message will be displayed:

```
"MSG1YN6t Not Enough Pfixed Getvis Storage to Establish Virtual Tape"
```

Installing the Virtual Tape Server

Note: The Virtual Tape Server is required for **Remote Virtual Tapes** only.

It must be installed on a workstation with a Java platform. The following steps are required:
- Obtain a copy of the Virtual Tape Server.
- Perform the Virtual Tape Server installation.

Support for uninstalling the Virtual Tape Server is also available.

The Virtual Tape Server is supplied as file **vtapesrv.w** (contained in PRD1.BASE) and also as file **vtape310.zip** (on the z/VSE Home Page). For details of how to obtain these files, see "Obtaining Copies of the JDK and Virtual Tape Server."

Obtaining Copies of the JDK and Virtual Tape Server

Before you begin, you must already have installed the *Java Development Kit* (JDK) 1.4 or higher on the workstation where you plan to install the Virtual Tape Server. You can download it from the Internet via the following URLs:

```
http://www.ibm.com/developerworks/java/jdk/index.html
(or)
http://java.sun.com/downloads/index.html
```

To obtain a copy of the *Virtual Tape Server*, get it either from the Internet or from the z/VSE library PRD1.BASE.

To obtain the Virtual Tape Server from the Internet, enter the following URL and download the file **vtape310.zip** (or **vtape310.zip-***APAR_number* for the latest APAR level) to the directory where you want to install the Virtual Tape Server.

```
http://www.ibm.com/servers/eserver/zseries/zvse/downloads/
```

To obtain the Virtual Tape Server from the z/VSE library PRD1.BASE, use the FTP (file transfer program) utility of TCP/IP for VSE/ESA to download the file **vtapesrv.w** to the directory where you want to install the Virtual Tape Server.

Notes:

1. You must download **vtapesrv.w** in *binary*.
2. Make sure that Unix mode is ***turned off***. Otherwise **vtapesrv.w** will be downloaded in ASCII mode, even when you specify *binary*. *UNIX mode* is one parameter of your VSE FTP daemon. Some FTP clients might *force* UNIX mode to be turned on. The example below shows how a successful transfer of **vtapesrv.w** was made using a (command line) FTP client. The place where the UNIX mode is set, is shown as bold.

```
c:\temp>ftp n.n.n.n   (this is the IP address of the z/VSE system)
Connected to n.n.n.n
220-TCP/IP for VSE -- Version 01.04.00 -- FTP Daemon
    Copyright (c) 1995,2000 Connectivity Systems Incorporated
220 Service ready for new user.
User (n.n.n.n:(none)): user
331 User name okay, need password.
Password:
230 User logged in, proceed.
ftp> cd prd1
250 Requested file action okay, completed.
ftp> cd base
250 Requested file action okay, completed.
ftp> binary
200 Command okay.
ftp> get vtapesrv.w
200 Command okay.
150-File: PRD1.BASE.VTAPESRV.W
    Type: Binary Recfm: FB Lrecl:    80 Blksize:      80
    CC=ON  UNIX=OFF RECLF=OFF TRCC=OFF CRLF=ON  NAT=NO
150 File status okay; about to open data connection
226-Bytes sent:      4,756,400
    Records sent:       59,455
    Transfer Seconds:       16.52 (    290K/Sec)
    File I/O Seconds:        3.94 (  1,548K/Sec)
226 Closing data connection.
4756400 bytes received in 17,12 seconds (277,91 Kbytes/sec)
ftp> bye
221 Service closing control connection.
c:\temp>ren vtapesrv.w vtape310.zip
```

Performing the Virtual Tape Server Installation

To perform the installation of the Virtual Tape Server, you must:

1. Open a Command Prompt and change to the directory into which you have downloaded the **vtape310.zip** file.
2. If you have downloaded the file from the z/VSE host, rename **vtapesrv.w** to **vtape310.zip**.
3. Extract **vtape310.zip** into a temporary directory (for example, `c:\temp`).
4. Execute the file:
   - **install.bat** or **install.cmd** (under Windows)
   - **install.sh** (under Linux or Unix)

   The installation process now begins, and you are guided through various installation menus.
5. The *Select Components to Install* window (Figure 70 on page 268) provides you with a choice:
   - Select *Virtual Tape Server* to install the Java code of the server.

- Select *Online Documentation* for online help and support information.

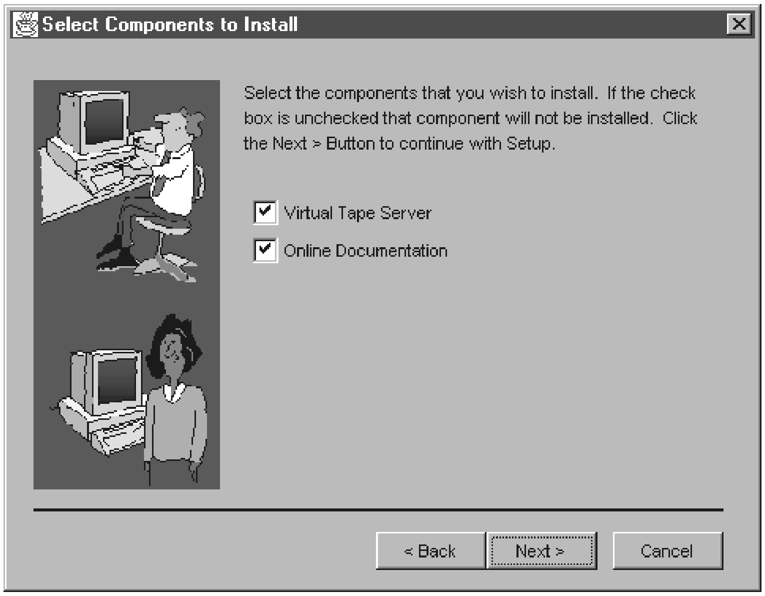


Figure 70. Selecting the Components of the Virtual Tape Server

After making your selection, click **Next**.

6. Select the directory you want to use. The default is **c:\vtape**.

   After making your selection, click **Finish** to complete the installation.

   Note: Desktop icons are created for Windows and OS/2.

Uninstalling the Virtual Tape Server

If there is a need to uninstall the Virtual Tape Server from your workstation, three methods are available:

- Using the **Uninstall** option provided by the Virtual Tape Server. From the Windows **Start**, select **Programs — Virtual Tape — Uninstall**, and the *Uninstall* window is displayed. Then select the components you wish to uninstall. If you select **Remove All Components**, after all components have been uninstalled, the icons on your desktop will also be removed.

  Note: For Windows, it is recommended to use this method.
- Running the batch uninstall program **juninst** manually. Using this method, you can *only uninstall all components.* Your desktop icons will also be removed.
- A third method is available under Windows only. From the Windows **Start**, select **Settings — Control Panel — Add/Remove Programs**. Now you can select **VSE Virtual Tape Server** and click **Install/Uninstall** to remove the Virtual Tape Server.

Starting the Virtual Tape Server

To start the Virtual Tape Server you may either click on the icon **Start Server** in the Program Group **Virtual Tape** (available on Windows only), or execute one of the script files **run.bat** or **run.cmd** (Windows) or **run.sh** (Linux or UNIX). The run script adds the *VirtualTape.jar* to the classpath and starts the Virtual Tape Server:

```
set classpath=.;VirtualTape.jar;%classpath%
java com.ibm.vse.vtape.VirtualTapeServer
```

Defining the Tape Device

A virtual tape is added to the system like a physical tape (unit). It must have at least one **cuu** address. For example:

```
ADD cuu,3480
```

The Virtual Tape Support is available for tape devices with a device type code of **3480**, **3490**, or **3490E**.

Note: It is recommended to select for virtual tapes unique *cuu* numbers that are not used by physical tape units in order to avoid any tape handling or tape operation problems.

Starting, Stopping, and Cancelling Virtual Tapes

You start and stop the Virtual Tape Support using the **VTAPE START** and the **VTAPE STOP** command. Examples are shown in the following sections and a detailed description of the command syntax and the command parameters is provided under "The VTAPE Command/Statement" on page 277.

The commands can be issued from both, a static or a dynamic partition, and also from a REXX procedure via the JCL host command environment (ADDRESS JCL).

Every job that contains a `VTAPE START,UNIT=`*`cuu`*`...` statement should contain an `ON $CANCEL GOTO` *`label`* statement, where *label* references the `VTAPE STOP,UNIT=`*`cuu`* command. By using this convention, you ensure that virtual tapes will *always* be closed at the end of the job, even if the job is cancelled.

Cancelling an I/O Operation to a Virtual Tape:
If you issue a `CANCEL cuu,FORCE` command where the cuu is being used to access a remote virtual tape, it can occur that the cuu can no longer be used to access the virtual tape. To be able to reuse the cuu, you should:

1. Close all virtual tapes as soon as possible using the `VTAPE STOP` command.
2. If the Tape Data Handler does not terminate automatically after 30 seconds, cancel the Tape Data Handler subsystem by issuing (where the Tape Data Handler is running in R1) a `CANCEL R1` command.
3. Issue a `VTAPE STOP` command for the cuu whose I/O has been cancelled.
4. Using the AR `VOLUME` command, check that no more virtual tapes are defined. A tape device type of `VTAP-00` in the command output indicates that the device is a virtual tape.
5. If there are still virtual tapes defined, you should issue a `VTAPE STOP` command for the remaining virtual tapes.

You can now redefine the virtual tapes using the `VTAPE START` command.

Starting and Stopping the Virtual Tape Data Handler

This section provides some background information on the startup of the Virtual Tape Data Handler. The Virtual Tape Data Handler is started automatically when the first VTAPE START command or statement is submitted. This causes the startup job TAPESRVR, stored in the VSE/POWER reader queue, to be released. By default, the Virtual Tape Data Handler runs in a **dynamic partition** of **class R**.

Note: You should view the Virtual Tape Data Handler as a subsystem of your z/VSE system. Therefore, if you wish to terminate virtual tape processing, you should **not** cancel the Virtual Tape Data Handler.

The Virtual Tape Data Handler requires the C-Runtime Library.

The startup job TAPESRVR is placed in the VSE/POWER reader queue either:
- during initial installation of z/VSE.
- during a cold startup of z/VSE.

z/VSE provides in VSE/ICCF library 59 skeleton SKVTASTJ which includes startup job TAPESRVR (Figure 71).

```
* $$ JOB JNM=CATSTVTA,DISP=D,CLASS=0
// JOB CATSTVTA       CATALOG TAPESRVR AND LDVTA, LOAD TAPESRVR
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG  TAPESRVR.Z        REPLACE=YES
// JOB TAPESRVR START UP VSE TAPE SERVER
// ID USER=VCSRV
// OPTION SYSPARM='00'
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// EXEC $VTMAIN,SIZE=$VTMAIN
$$/*
$$/&
$$$$ EOJ
/+
CATALOG  LDVTA.PROC     REPLACE=YES DATA=YES
// EXEC DTRIINIT
   LOAD TAPESRVR.Z
/*
/+
/*
// EXEC PROC=LDVTA        TO LOAD  TAPE SERVER  INTO RDR QUEUE
/&
* $$ EOJ
```

Figure 71. Skeleton SKVTASTJ (for Starting the Virtual Tape Data Handler)

The skeleton does the following:
1. Catalogs startup job as TAPESRVR.Z into IJSYSRS.SYSLIB.
2. Catalogs procedure LDVTA which loads TAPESRVR.Z into the reader queue via DTRIINIT.
3. Executes LDVTA.PROC. If you do not want to load the startup job immediately into the active reader queue, delete the line // EXEC PROC=LDVTA.

The job stream loads the Virtual Tape Data Handler into a dynamic partition of class R which occupies 8 MB. 8 MB is the minimum required.

If you create a new TAPESRVR job, you must ensure that this new job is released by the VTAPE START command. To do so, delete the previous TAPESRVR job in the VSE/POWER reader queue.

Note: You cannot change the job name TAPESRVR.

Here are some guidelines for stopping the Virtual Tape Data Handler:
- If you want to terminate the Tape Data Handler subsystem, use the `VTAPE STOP` command to close all virtual tapes. The Tape Data Handler will then automatically terminate after 30 seconds.
- To check which tape units are defined as virtual, use the AR (attention routine) `VOLUME` command.
- After you terminate the Tape Data Handler, you should check to see if any virtual tapes are still active. If there are virtual tapes that are active, issue a `VTAPE STOP` command against these virtual tapes.
- If the Tape Data Handler is cancelled because of an error condition (for example, a program check), to avoid possible inconsistencies in the system you should close all open virtual tapes. To see which virtual tapes are open, use the AR `VOLUME` command. A tape device type of `VTAP-00` in the command output indicates that the device is a virtual tape.

Working with VSE/VSAM Virtual Tapes

To work with VSE/VSAM virtual tapes, you must first define the VSE/VSAM ESDS file which is to contain the virtual tape. This can be done via either the:
- Dialog *Define a New File* (Fastpath 222).
- VSE/VSAM IDCAMS job stream provided in skeleton SKVTAPE (which is available in VSE/ICCF library 59).

You can then use the JCL command VTAPE to open the VSE/VSAM file as a virtual tape which can be accessed via the *cuu* specified. For example:

```
VTAPE START,UNIT=cuu,LOC=VSAM,FILE='vsamfilename',SCRATCH
```

In the example, the SCRATCH parameter is optional and causes the virtual tape to be cleared before new data is written to it.

```
VTAPE STOP,UNIT=cuu
```

closes the virtual tape file and drops the association between the tape unit *cuu* and the virtual tape file. If this was the only active VTAPE session, the z/VSE partition in which the Virtual Tape Data Handler is running is released after 30 seconds.

VSE/VSAM ESDS File Definition (Skeleton SKVTAPE)

Skeleton SKVTAPE creates a VSE/VSAM ESDS file that is to contain a virtual tape. Figure 72 shows the skeleton.

```
* $$ JOB JNM=SKVTAPE,CLASS=0,DISP=D
// JOB SKVTAPE     CREATE VIRTUAL TAPE FILE
// EXEC IDCAMS,SIZE=AUTO
   DELETE (VSE.VTAPE.FILE) PURGE CL -
          CATALOG(VSESP.USER.CATALOG)
   DEFINE CLUSTER (NAME(VSE.VTAPE.FILE) -
          RECORDS (1000 1000) -
          TO (99366) -
          REUSE -
          NONINDEXED -
          SHAREOPTIONS (1) -
          SPANNED -
          RECORDSIZE (32758 32758) -
          VOLUMES (-V001-)) -
          DATA (NAME(VSE.VTAPE.FILE.@D@) -
          CISZ (32768)) -
          CATALOG (VSESP.USER.CATALOG)
    IF LASTCC NE 0 THEN CANCEL JOB
/*
// OPTION STDLABEL=DELETE
VTAPE1
/*
// OPTION STDLABEL=ADD
// DLBL VTAPE1,'VSE.VTAPE.FILE',99/366,VSAM,CAT=VSESPUC
/*
// EXEC IESVCLUP,SIZE=AUTO             ADD LABEL TO STDLABUP PROC
D                                              VTAPE1
A VSE.VTAPE.FILE                               VTAPE1  VSESPUC
/*
/&
* $$ EOJ
```

Figure 72. Skeleton SKVTAPE

Explanations to skeleton SKVTAPE:

-V001-
: This variable defines the volume on which the VSE/VSAM virtual tape file is located. The VSAM space is part of the VSE/VSAM user catalog VSESPUC (VSESP.USER.CATALOG).

RECORDS
: The number of RECORDS depends on the amount of data the virtual tape is to contain. The definition provided in SKVTAPE results in a file size of about 32MB. If this size does not meet your requirements, modify the definitions in SKVTAPE accordingly.

REUSE
: REUSE allows repeated writing to a virtual tape from the beginning. When specifying the SCRATCH parameter in the VTAPE START command, the tape is cleared before new data is written to it. Without the SCRATCH parameter, existing data is overwritten. Refer also to "Writing to VSE/VSAM Virtual Tapes" on page 273.

: NOREUSE is for creating a new and empty file for writing data to it once; for backup, for example. NOREUSE together with SCRATCH is invalid and causes an error message.

SHAREOPTIONS
: SHAREOPTIONS (1) allows multiple READ operations or one

WRITE operation to be active. It is recommended not to change the setting of the share option. Refer to "File Sharing" for further details.

RECORDSIZE
The value for the RECORDSIZE parameter is the CISZ value minus 10 bytes (CISZ minus 20 bytes in case of a compressed cluster).

CISZ
The recommended value for the Control Interval Size is 32 KB; it should not be smaller than 8 KB.

File Size

The 4GB limit for VSE/VSAM files applies also for VSE/VSAM virtual tape files.

File Name

In the skeleton, **VTAPE1** in the DLBL statement is the filename to be used in the VTAPE START command.

File Sharing

Whatever is defined in SHAREOPTIONS, the Virtual Tape Data Handler accepts only single WRITE **or** multiple READ access to a VSE/VSAM virtual tape from a single z/VSE system. This corresponds to SHAREOPTIONS (1). If you want to access a VSE/VSAM virtual tape from more than one z/VSE system, use SHAREOPTIONS (1) which allows single WRITE **or** multiple READ access only. SHAREOPTIONS (1) avoids unpredictable results where multiple system access is being used.

Writing to VSE/VSAM Virtual Tapes

WRITE can only start at the beginning of a VSE/VSAM virtual tape. No rewrite or overwrite of existing data is possible except from the beginning a tape. For writing to a VSE/VSAM virtual tape more than once, the file definition (skeleton SKVTAPE) must include the REUSE parameter. Once a VTAPE STOP command has been issued, the remaining buffer contents is written to tape, the tape is closed, and an end-of-volume (EOV) indicator is written. This means no further data can be appended (added) to this particular virtual tape.

There are three ways of writing to a VSE/VSAM virtual tape:

1. The VSE/VSAM file is defined with REUSE.

   During OPEN processing the tape is positioned to its beginning before a WRITE operation starts. Existing data is overwritten.
2. The VSE/VSAM file is defined with REUSE and VTAPE START includes the SCRATCH parameter.

   SCRATCH causes the tape to be cleared before WRITE starts at the beginning of the tape.
3. The VSE/VSAM file is defined with NOREUSE.

   You can write to the tape only once (for backup, for example).

   NOREUSE with SCRATCH is invalid and causes an error.

Working with Remote Virtual Tapes

To access a remote virtual tape, a TCP/IP connection must be established between z/VSE and the remote workstation with the **Virtual Tape Server** installed. You start the VTAPE support for a remote virtual tape with a VTAPE command such as the following:

```
 VTAPE START,UNIT=cuu,LOC=ipaddress:portnumber,FILE='filename'
(or)
 VTAPE START,UNIT=cuu,LOC=hostname:portnumber,FILE='filename'
```

The *ipaddress* or *hostname* identifies the Virtual Tape Server workstation on which the remote virtual tape file is located (a Linux, UNIX, or Windows file, for example). The *portnumber* is the TCP/IP port number to be used for the connection. If the virtual tape file does not yest exist, the command automatically causes the creation of the required file (Linux, UNIX, or Windows) using the *filename* specified. The VTAPE command:

```
 VTAPE STOP,UNIT=cuu
```

drops the association between the tape unit *cuu* and the remote virtual tape file. If this was the only active VTAPE session, the z/VSE partition in which the Virtual Tape Data Handler is running is released after 30 seconds.

Note: If a remote virtual tape has been started whose TCP/IP connection is slow, this might have a negative impact on the performance of other virtual tapes. To avoid this problem, you should ensure that the TCP/IP connection to the virtual tape is reliable and fast enough to be able to process high data-transfer rates.

File Names and other Considerations

If the required Linux, UNIX, or Windows file for a remote virtual tape does not exist yet, it is automatically created when submitting the corresponding VTAPE START command. When assigning file names you must observe certain rules and characteristics as outlined below.

File Name Considerations

File names may contain blanks, therefore the *filename* must be enclosed in quotes. A quote within a *filename* must be coded as two single quotes; for example:

```
 FILE='D:\John''s\Virtual Tapes\vt001401.001'
```

File *names* can have more than 100 characters in length, therefore you may specify `FILE='filename'` twice or even three times. However, each file *parameter* cannot be greater than 100 characters in length. Therefore, the *filename* is concatenated in storage, allowing for a file name length of 200 or even 300. The following example is equivalent to the previous example:

```
 FILE='D:',FILE='\John''s\Virtual Tapes\',FILE='vt001401.001'
```

Linux and UNIX Considerations

Linux and UNIX are case sensitive but job streams created on the z/VSE host (dialogs of the Interactive Interface) are in capital letters. It may be therefore necessary to edit such job streams to adapt the *filename* to the Linux or UNIX conventions.

Windows Considerations

You can use forward slashes (“/”) instead of backward slashes (“\”) on Windows (as is the case with Linux and UNIX).

Note: To avoid problems that might occur during the code page translation of backward slashes, you are recommended to use forward slashes under Windows.

The Virtual Tape Server can accept both forward and backward slashes on Windows.

Further Documentation

The workstation with the Virtual Tape Server installed provides the document **vtape.html**. It includes further details about remote virtual tapes and the related environment. You reach the document by selecting from the Start Menu first *Virtual Tape*, and then *Help.*

Examples of Using Virtual Tapes

Backing Up and Restoring Data

Backup functions often produce tape output. Thus you may consider whether virtual tapes are an alternative for such tasks in your data processing environment. For temporary data or short term backups in a z/VSE environment, for example, VSE/VSAM virtual tapes avoid the overhead of mounting and handling physical tapes.

Following is a backup example using a remote virtual tape. Data on a remote virtual tape can be saved on CD-ROM for archiving or for distribution. In the example, a VSE library backup and a later restore is assumed.

1. Create a job stream to backup a VSE library on a remote virtual tape as shown in the following sample job stream:

```
// JOB BACKUP (Backup a Library to a PC File)
// ON $CANCEL OR $ABEND GOTO VTAPSTOP
VTAPE START,UNIT=480,LOC=9.164.186.20:2285,                          C
               FILE='D:\VSE Backup\prd2.001'
MTC REW,480
// EXEC LIBR
   BACKUP LIB=PRD2 TAPE=480
/*
/. VTAPSTOP
VTAPE STOP,UNIT=480
/&
```

Figure 73. Job Stream Example for Backing Up a z/VSE Library

2. Use the programs available on your Linux or Windows system to copy (burn) the VSE library from the remote virtual tape to a CD-ROM for archiving.

   Depending on the CD-ROM software you have installed, you may create the virtual tape backup of the VSE library directly on the CD-ROM.
3. To restore the VSE library you must create a similar job stream as the one shown for BACKUP, using the Librarian RESTORE command. You can restore the VSE library (stored as virtual tape on CD-ROM) directly from the CD-ROM back to z/VSE.

Transferring Virtual Tape Files

It is possible to transfer virtual tape files between a workstation (remote virtual tape) and the z/VSE host (VSE/VSAM virtual tape) and vice versa with the

TCP/IP File Transfer Program (FTP). For a transfer from workstation to host use the **put** command, for a host to workstation transfer the **get** command. Make sure you are transferring the files in **binary** mode.

The example below shows the command sequence for a transfer from server to host:

```
C:\>ftp x.x.x.x                     <-- enter hostname/IP address of your VSE system
Connected to x.x.x.x.
220-TCP/IP for VSE -- Version 01.04.00 -- FTP Daemon
    Copyright (c) 1995,2001 Connectivity Systems Incorporated
220 Service ready for new user.
User (9.164.155.2:(none)): user      <-- enter your user id here
331 User name okay, need password.
Password:                            <-- enter your password here
230 User logged in, proceed.
ftp> bin                             <-- switch to binary mode
200 Command okay.
ftp> quote site lrecl 32758          <-- enter the record size of your VSAM cluster
200 Command okay.
ftp> quote site recfm v              <-- set record format to variable
200 Command okay.
ftp> put tape.image VSE.VTAPE.FILE   <-- enter your filenames
```

Installing Optional Programs

The z/VSE dialog for installing optional programs supports virtual tapes. It is thus ready for use when an optional program is available for electronic delivery. The program can then be downloaded to a virtual tape file from where it can be installed via the dialog as from any other physical tape containing optional programs.

For details about how to install optional programs using virtual tapes, refer to the manual *z/VSE Installation*, SC33-8222.

The VTAPE Command/Statement

You start and stop the z/VSE Virtual Tape Support through the **VTAPE START** and the **VTAPE STOP** command or statement.

Figure 74 shows the syntax of the VTAPE START and VTAPE STOP command/statement.

```
>>──┬────┬──VTAPE START,UNIT=cuu,LOC=──┬─VSAM──────────────────────────────┬──────────►
    └─// ┘                             │                ┌─:2386───────┐    │
                                       └─┬─ipaddress─┬──┼─────────────┼────┘
                                         └─hostname──┘  └─:portnumber─┘

   ┌──────────────────────────┐
   ▼                      (1) │  ┌─,WRITE───┐
►────,FILE='filename' ────────┴──┼──────────┼──────────────────────────────────────►◄
                                 ├─,SCRATCH─┤
                                 └─,READ────┘
```

Notes:

1 FILE='*filename*' can be specified up to three times.

```
>>──┬────┬──VTAPE STOP,UNIT=cuu──────────────────────────────────────────────────►◄
    └─// ┘
```

Figure 74. Syntax of VTAPE START and VTAPE STOP Command/Statement

The operands have the following meaning:

START
Indicates that a tape unit is to be associated with a virtual tape file. In case of LOC=*ipaddress* or LOC=*hostname*, a TCP/IP connection to the Virtual Tape Server workstation is established.

STOP
Indicates that an existing association between a tape unit and a virtual tape file is to be dropped. In case LOC=*ipaddress* or LOC=*hostname* was specified in the corresponding START command, the TCP/IP connection to the workstation is closed.

UNIT=cuu
Specifies the tape unit to be used for the virtual tape. The *cuu* must be added to the system like a physical tape unit with the IPL ADD command. For example:

```
ADD cuu,3480
```

Valid device type codes are 3480, 3490, and 3490E.

LOC=VSAM
Indicates that the virtual tape resides in a VSE/VSAM ESDS file on the z/VSE system.

LOC=ipaddress:portnumber (or) LOC=hostname:portnumber
Indicates that the remote virtual tape resides on a remote system (Linux, UNIX, or Windows, for example) located on the Virtual Tape Server workstation identified by the *ipaddress* or *hostname*. The *portnumber* specifies the TCP/IP port number to be used for the connection. If this operand is omitted, the default port number of 2386 is taken.

FILE='filename'
Identifies the file which contains the virtual tape.

- For a **VSE/VSAM file**, *filename* is the 1 to 7 alphameric-characters filename of the DLBL statement.
- For a **remote file**, the *filename* can be up to 100 characters in length, enclosed in quotes. The Linux, UNIX, and Windows conventions for file names apply. Refer to "File Names and other Considerations" on page 274 for further details.

READ
Specifies that read access to the virtual tape is required.

WRITE
Specifies that write access to the virtual tape is required. This is the default. WRITE includes READ access.

When writing to **VSE/VSAM virtual tapes**, certain rules and restrictions exist. Refer to "Writing to VSE/VSAM Virtual Tapes" on page 273 for details.

SCRATCH
Specifies that write access to the virtual tape is required and that the tape is to be cleared first. It requires REUSE in the VSE/VSAM file definition (skeleton SKVTAPE). SCRATCH includes WRITE and READ access.

Restrictions When Using Virtual Tapes

In general, the Virtual Tape Support is intended to be transparent to applications, and to provide customers with the ability to read from or write to a virtual tape in the same way as if it were a physical tape. For technical and performance reasons, the full range of the capabilities of a physical tape has not been implemented and there are a number of restrictions outlined in this section. This section also provides and points to planning information to be considered before you start using virtual tapes at your installation.

The following restrictions apply for virtual tapes:

- There is no alternate tape support for virtual tapes. To avoid this restriction:
  - For a VSAM virtual tape, you should choose primary and secondary allocations for the virtual tape file that are large enough to avoid an "end of volume" condition.
  - For a remote virtual tape, the size of the remote virtual tape file is assigned according to the PC's file rules. If you receive an "end of volume" condition for your remote virtual tape, you should check these PC file rules.
- The SDAID trace program does not support virtual tapes.
- The DITTO/ESA for VSE program does not support the ERASE TAPE function for virtual tapes. An application that issues an ERASE TAPE function will be cancelled. Instead of using the ERASE TAPE function, you should use the equivalent VSAM and PC file-system functions.
- Multivolume tape files and alternate tapes (ALT option of the ASSGN statement) are not supported for virtual tapes.
- Virtual Tape Support in a standalone environment is not supported.
- You are recommended **not** to run virtual tape job streams in VSE/ICCF interactive partitions.
- For remote virtual tapes, the file you need for containing a virtual tape is created *automatically*. However, for a VSE/VSAM virtual tape you must *create the file yourself*. For details, see "VSE/VSAM ESDS File Definition (Skeleton SKVTAPE)" on page 272.

- A remote virtual tape behaves like a physical tape with regard to reading/writing. However, a VSE/VSAM virtual tape behaves differently with regard to writing. For details, see "Writing to VSE/VSAM Virtual Tapes" on page 273.

Restrictions Regarding Label Handling:
The following job sequence would cause a deadlock between the partition BG and the Tape Data Handler partition:

```
// OPTION STDLABEL=ADD
// DLBL VTAPFIL,'VTAPFIL.TEST.FILE',,VSAM,CAT=IJSYSCT
// VTAPE START,UNIT=181,LOC=VSAM,FILE='VTAPFIL'
```

The deadlock is caused by the VTAPE command, since the BG partition is concurrently updating system standard labels (that is the `// OPTION STDLABEL` is in effect). To avoid this type of deadlock, you should terminate the standard label processing (by including a `// OPTION USRLABEL` statement before the `VTAPE START` statement).

The correct job control sequence would therefore be:

```
// OPTION STDLABEL=ADD
// DLBL VTAPFIL,'VTAPFIL.TEST.FILE',,VSAM,CAT=IJSYSCT
// OPTION USRLABEL
// VTAPE START,UNIT=181,LOC=VSAM,FILE='VTAPFIL'
```

For the latest information on supported functions and programs, operating characteristics, and possible restrictions refer also to the z/VSE home page (Service and Support page) at:

`http://www.ibm.com/servers/eserver/zseries/zvse/support/`

Chapter 14. Implementing Native 3494 Tape Library Support

Previous to z/VSE 3.1, the 3494 Tape Library dataserver could be accessed:

- From z/VM, using the DFSMS interface (VSE Guest Server support).
- Under VSE using the LCDD (Library Control Device Driver).

From z/VSE 3.1 onwards, the z/VSE *Tape Library Support* (TLS) enables z/VSE to be accessed via the System/390 channel interface. As a result:

- The need to use an XPCC or APPC communication protocol is removed.
- The need to setup a VTAM LU6.2 connection when using LCCD is removed.

From z/VSE 3.1 onwards, using this "native" 3494 TLS the IBM TotalStorage 3494 Virtual Tape Server (3494 VTS) can also be used.

This chapter describes the z/VSE Tape Library Support in these main sections:

- "Migrating/Configuring Your z/VSE System for TLS"
- "Understanding the Format of Inventory Data" on page 283
- "Performing Tape Library Functions" on page 285

Related Sections:

- For details about the LBSERV macro, refer to the manual *z/VSE System Macros Reference*, SC33-8230.
- For details about LIBSERV JCL, refer to the manual *z/VSE System Control Statements*, SC33-8225.

Migrating/Configuring Your z/VSE System for TLS

To prepare your z/VSE system for use with an IBM 3494 tape library, you must perform the steps described below:

Step 1. **Add a SYS ATL=TLS Command to Your z/VSE Startup Procedure:**
You must add a `SYS ATL=TLS` command to your z/VSE IPL procedure, using the dialog described in "IPL Parameters You Can Modify" on page 9. For details of the SYS command, refer to the manual *z/VSE System Control Statements*, SC33-8225.

Step 2. **Define Customization Options and Control Statements:**
You must change the sample job TLSDEF which is provided in ICCF Library 59 to meet your own system requirements. Here is an example job which uses logical addresses 460 to 463, and 580 to 582. All variables are shown in italics.

```
* $$ JOB JNM=TLSDEF,CLASS=0,DISP=D
* $$ LST CLASS=A
// JOB TLSDEF
   EXEC LIBR,PARM='MSHP'
   ACCESS S=IJSYSRS.SYSLIB
   CATALOG TLSDEF.PROC REPLACE=YES
   LIBRARY_ID TAPELIB1 SCRDEF=SCRATCH00 INSERT=SCRATCH00
   LIBRARY_ID TAPELIB2                * SECOND LIB DEF DUAL LIB
   DEVICE_LIST TAPELIB1 460:463       *  DRIVES  460 TO 463
   DEVICE_LIST TAPELIB2 580:582       *  DRIVES  580 TO 582
   QUERY_INV_LISTS LIB=TLSINV         * MASTER INVENTORY FILES
```



```
   MANAGE_INV_LISTS LIB=TLSMAN         * MANAGE FROM MASTER
/+
/&
* $$ EOJ
```

These are the keywords and parameters used in job TLSDEF:

LIBRARY_ID
The keyword LIBRARY_ID is followed by the eight-character logical unit name of an attached library. This logical unit name is used as the Library Name field in functional requests from users, and as the z/VSE sub-library name used in inventory requests. If the library name is less than 8 characters, it will be padded to the right with blanks.

- The keyword SCRDEF is followed by the name of the scratch pool to be used as the default for this z/VSE host on this library. Tapes will be mounted from this pool for nonspecific MOUNT SCRATCH requests. If this parameter is not included, SCRATCH00 will be used as the default pool.
- The keyword INSERT allows a target category to be specified for automatic insert processing of new volumes inserted in the library. The target category must be either SCRATCH*nn* (where nn is in the range 00 to 31), or PRIVATE. If this parameter is omitted, no automatic insert processing occurs.

You must define one LIBRARY_ID statement for *each* attached library. If user requests do not specify a library name, the library associated with the first occurrence of this keyword will be used as the default. z/VSE supports a maximum of *eight* libraries.

DEVICE_LIST
Designates the Library_Id and the corresponding drives. When multiple hosts are attached to the same library, this keyword allows device partitioning. This statement may be repeated as required to list all libraries with drives to be used by this host.

QUERY_INV_LISTS
Designates the name (up to seven characters) for the predefined VSE sub-library in which Query Inventory member lists are to be created. If this control card is not found or if it is not coded properly, or if the library and a sub-library for the attached 3494 is not defined, Query Inventory requests cannot be processed.

MANAGE_INV_LISTS
Designates the name (up to seven characters) for the predefined library from which Manage Inventory member lists are to be read. If this control card is not found or if it is not coded properly, or if the library and a sub-library for the attached 3494 is not defined, Manage Inventory requests cannot be processed.

For details of how to predefine z/VSE libraries for inventory requests, refer to Step 4 (below).

After you have customized the job TLSDEF, submit this job to catalog TLSDEF.PROC into library IJSYSRS.SYSLIB. Your TLSDEF.PROC will then become active the next time an IPL is made of your z/VSE system.

Step 3. **Enter a STDEVOPT Statement in z/VM (Optional)**
If your z/VSE system is running under z/VM, you must include this statement in the user directory entry:

```
STDEVOPT LIBRARY CTL
```

Otherwise, the drives will be controlled by z/VM which will return a "command reject" for each request.

Step 4. **Define z/VSE Libraries to Contain Inventory Requests**
Before you can start the inventory application, you must create a library into which host copies of 3494 inventory lists can be written. You must create a unique sub-library that has the name of your 3494 Tape Library.

The library into which host copies of 3494 inventory lists can be written requires disk space assigned using DLBL and EXTENT statements. You can add these labels to either the STANDARD or USER STANDARD LABEL procedure. Here is an example of these statements:

```
* DEFINITION FOR THE VSE TLSINV INVENTORY LIBRARY
// DLBL TLSINV,'VSE.TLSINV.QLISTS.LIBRARY',99/365
// EXTENT ,SYSWK1,1,0,9500,100
EXEC LIBR
DEF L=library
DEF S=library.libraryname
```

Understanding the Format of Inventory Data

Output File Produced by a Query Inventory Request

An 80-byte inventory record output file is created by a Query Inventory request. This file applies to a cartridge and contains the following information:

- External volume label (six characters)
- Media type:
  - CST1 or CST2 for 3490 cartridges.
  - CST3 or CST4 for 3590 cartridges.
  - CST5 for 3592 300 GB cartridges.
  - CST6 for 3592 WORM (Write-Once-Read-Many) 300 GB cartridges.
  - CST7 for 3592 60 GB cartridges.
  - CST8 for 3592 WORM 60 GB cartridges.
- Special attribute byte, represented by an EBCDIC bit string (eight characters)
  Bit 0 If 1, volume is present in library, but inaccessible
  Bit 1 If 1, volume is mounted or queued for mount
  Bit 2 If 1, volume is in eject-pending state
  Bit 3 If 1, volume is in process of ejection
  Bit 4 If 1, volume is misplaced
  Bit 5 If 1, volume has unreadable label or no label
  Bit 6 If 1, volume was used during manual mode
  Bit 7 If 1, volume was manually ejected
- Category name (ten characters)
- Library manager hexadecimal category number, in EBCDIC representation (four characters)

The fields in each record are separated by a blank character to enhance their readability.

A sample file record is:

```
CS0010 CST2 01000000 PRIVATE    FFFF
```

A header with the time of list creation is inserted as the first record in the list.

Input File Submitted by a Manage Inventory Request

A file submitted for use in a Manage Inventory request requires that each six-character external volume serial number in the list start in column 1 of a file record. The remaining space in each 80-character record is ignored as input. This allows for returning the Query Inventory output file as input to the Manage Inventory function.

A sample record in a Manage Inventory input file is:

```
CS0010 CST2 01000000 PRIVATE    FFFF
```

Or, simply

```
CS0010
```

A header record (as described for Query Inventory output) may be optionally present in the input list. Any record starting with an asterisk (*) is not considered a valid input data record and is ignored.

Output Produced by a Manage Inventory Request

After a Manage Inventory request is completed, a return code (or reason code) is supplied to indicate that processing is complete and to report the overall results for processing the request (for example, input was valid, file was found, and so forth). The actual outcome of transferring each volume to a new target category is reflected within the file itself. The file is updated by adding a results message in each file record, starting in column 38. An example of a successful output file record is:

```
CS0010 CST2 01000000 PRIVATE    FFFF    CATEGORY CHANGED TO EJECT
```

An example for an unsuccessful output file record is:

```
AB1234     CATEGORY NOT CHANGED, RSN=3340
```

Naming Conventions for Inventory Files

Table 6 summarizes the naming conventions for inventory files used in the Query Inventory (IQUERY) functions. VSE libraries must be predefined, together with a sub-library for each attached 3494.

Table 6. Naming Conventions for Inventory Files

Source Category	Member Name
(omitted)	ALL
PRIVATE	PRIVATE
INSERT	INSERT
SCRATCHnn	SCRnn
SCRATCH	SCRnn (see Note 1)
MANEJECT	MANEJECT
EJECT	EJECT
VOL	VOL

Notes:

1. In the above table entry for SCRATCH, *nn* refers to the default scratch pool that was specified using the SCRDEF keyword (see Step 2 of "Migrating/Configuring Your z/VSE System for TLS" on page 281).

2. There are no naming conventions for the *Manage Inventory* (MINVENT) function. You can specify MEMNAME according to your own naming conventions.

Performing Tape Library Functions

IBM 3494 Tape Library communication is performed using the JCL LIBSERV statements described in Table 7. For details of these commands , refer to the manual *z/VSE System Control Statements*, SC33-8225.

Table 7. Overview of LIBSERV Commands Used With the IBM 3494 Tape Library Data Server

Command	Description
LIBSERV AQUERY	Verify the location of a volume in all known libraries.
LIBSERV CQUERY	Return the number of volumes in a source category.
LIBSERV CMOUNT	Mount a volume from a source category on a device belonging to a tape library. Access will then be given to the device.
LIBSERV DQUERY	Verify the status of a device in a tape library.
LIBSERV RELEASE	Release a device in a tape library.
LIBSERV EJECT	Eject a volume.
LIBSERV IQUERY	Request the inventory data of volumes currently assigned to a category in a tape library. See also "Output File Produced by a Query Inventory Request" on page 283.
LIBSERV LQUERY	Return the operational status of a tape library.
LIBSERV MOUNT	Mount a specified volume.
LIBSERV MINVENT	Transfer the volumes in a referenced member name and source category, to a target category.
LIBSERV SETVCAT	Assign a volume to a target category.
LIBSERV SQUERY	Verify the location of a volume in a library.

Chapter 15. Performing a FlashCopy Using the IXFP Command

This chapter describes how you can perform a FlashCopy using the AR (Attention Routine) command **IXFP**. It contains these main sections:

- "Installing FlashCopy"
- "Syntax and Parameters of the IXFP Command" on page 288
- "Issuing IXFP Commands From a Batch Job" on page 291
- "Using FlashCopy to Backup VSE/VSAM Datasets" on page 292
- "Using IXFP SNAP Function With VM Minidisks" on page 298
- "VSE/Fast Copy (FCOPY) Exploitation of FlashCopy" on page 298

Installing FlashCopy

Hardware Prerequisite

FlashCopy support has the following hardware prerequisites:

- IBM TotalStorage Enterprise Storage Server (ESS) with the FlashCopy Version 1 or Version 2 feature installed.
- IBM TotalStorage DS8000 or DS6000 with the PTC ("point-in-time copy") feature installed.

IBM TotalStorage FlashCopy technology is a point-in-time copy capability that can be used to help reduce planned application outages caused by backups and other data copy activities.

z/VSE 3.1 provides support for FlashCopy Version 1 (volume-based FlashCopy) including the NOCOPY option. The NOCOPY option can be used to copy most, or all, of the data directly from the source to tape *without* the need to first copy all of the physical data to an intermediate backup copy.

FlashCopy Version 2 ("FlashCopy 2") includes the features of FlashCopy (including NOCOPY) *plus* extensions designed to improve capacity management and disk utilization. z/VSE 3.1 supports the following selected FlashCopy Version 2 functions:

- *Dataset Copy*, which reduces background completion times because FlashCopy no longer needs to be performed at the volume level when only a dataset copy is required. Dataset Copy allows the source and target volumes may be different sizes, and the copied data to reside at a different location in the source and target volumes.
- *Elimination of the LSS Constraint* can help simplify administration and capacity planning for FlashCopy. Source and target volumes can now span logical subsystems within a storage server.
- *Multiple Relationship FlashCopy* offers new flexibility. It allows up to 12 target volumes to be created from one source volume in a single FlashCopy operation.
- *Persistent FlashCopy Relationship* allows the relationship to remain, even after the FlashCopy operation is complete.
- *Performance improvements* in FlashCopy 2 are designed to reduce the time required to complete a FlashCopy establish command.

z/VSE 3.1 does **not** support these FlashCopy functions:

- Incremental FlashCopy.
- Consistency Group commands over a Remote Mirror link.
- Inband Commands over a Remote Mirror link.

Shipment and Installation

The FlashCopy support is part of z/VSE as shipped. The name of the phase containing this support is $IJBESFC, which resides in sublibrary IJSYSRS.SYSLIB.

$IJBESFC is automatically loaded into the SVA during startup. The FlashCopy support is provided via the IXFP command as described under "Syntax and Parameters of the IXFP Command."

Syntax and Parameters of the IXFP Command

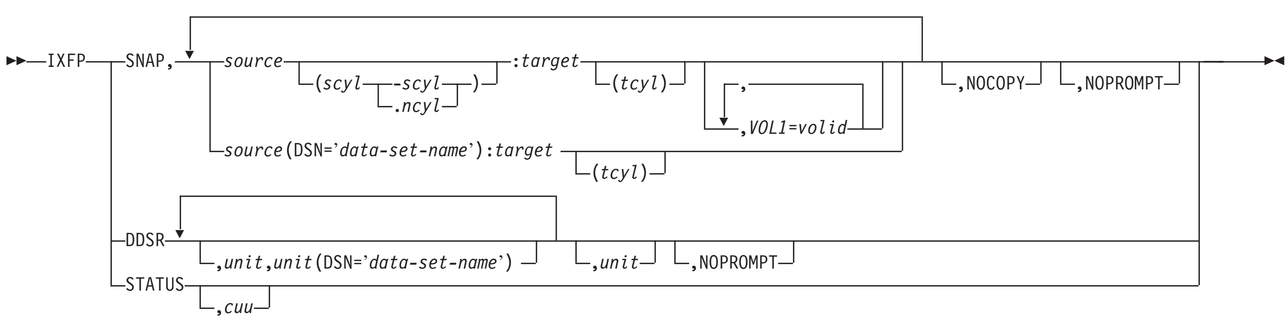
Using the IXFP **SNAP** function, the FlashCopy functionality is invoked. In addition, you can use the IXFP **DDSR** and **STATUS** functions, as described below.

Note: Except for FILE SNAPping (DSN=data-set-name), VSE will **not**:
1. perform VTOC checking on the specified target device.
2. provide warning messages of any kind, be it overlapping extents, secured- or unexpired files, or anything else.

Cylinder or volume copying will be done *unconditionally* within the specified or assumed boundaries.

IXFP SNAP copies data from a source device to a target device. Using this command, the FlashCopy functionality is invoked.

JCC, AR Format



SNAP The IXFP SNAP function copies data from a source device to a target device.

source The device ID (cuu) or the VOL1 label of the SOURCE device required when copying data from it onto a TARGET device. If the SOURCE device is identified by its VOLID, it must be either the only volume with that VOLID, or it must be the only VOLUME with that VOLID which is up (DVCUP), otherwise an error message will be issued. The whole VOLUME will normally be copied unless the operator has provided additional information that either identifies a cylinder range or a Data-Set-Name (DSN) contained on the source device that is to be copied.

scyl-scyl
The decimal start- and end-cylinder range where copying is to start and where it is to end on the source device. Cylinder is the smallest entity that

can be specified for any SNAP command function. The highest (end) cylinder number must not exceed the device's primary number of cylinders and the start cylinder number must not be higher than the end cylinder number.

scyl.ncyl
: The decimal start-cylinder where copying is to start on the source device and the number of cylinders (ncyl) that should be copied. Cylinder is the smallest entity that can be specified for any SNAP command function. The highest resulting cylinder number must not exceed the device's number of primary cylinders.

DSN= The data-set-name identifying the file on the source device, which must be a **non-VSAM** file, that the operator wants to be copied onto the target device. The file will be copied into the exact extent boundaries where it was located on the source device.

However, SAM (Sequential Access Method) files can be relocated (assuming that the level of the hardware support provides this function) to a different, single extent disk location on the target device. In this case, the **tcyl** operand must be supplied but the device must **not** be a **VM-partial minidisk**. The proper label information (single FORMAT-1 label) will be created and added to the target VTOC.

Processing multi-volume-files is the responsibility of the operator, such that the SNAP command should be repeated for all the source volumes containing file extents.

The number of cylinders to be copied is limited by the limits existing for the source device. Copying will only be performed if the appropriate extent boundaries on the TARGET device are available or have already expired, otherwise an error message will be provided. (Refer to the DDSR function in case the overlaid file should be deleted and released).

target The Device-ID (cuu) or the VOL1 label of the TARGET device is required when copying data to it from a SOURCE device.

The target device must be set DOWN (DVCDN command) prior to initiating the SNAP function, except the source and the target device are the same device (user is copying data from one location of a disk into another location on the same disk), or except a file (DSN=data-set-name) is being copied. If the TARGET device is identified by its VOLID, it must either be the only volume with that VOLID, or it must be the only VOLUME with that VOLID which is DOWN (DVCDN), otherwise an error message will be issued.

As many cylinders as allocated on the SOURCE device will be used for file copying onto the target device (DSN=data-set-name). Otherwise, as many cylinders as specified for the SOURCE device, or the whole SOURCE volume, will be copied onto the TARGET device.

If the specified cylinder range does not match the cylinder range that was given for the source device, relocation of data records will be assumed. This applies for **ESS** only, and providing the level of the hardware supports this function.

If the cylinder range does not match the cylinder range of the source device and the target device is a **VM partial-pack minidisk**, the command will be rejected. This is because VM uses virtual cylinder values for partial-pack minidisks, and the cylinder ranges must match for VM partial-pack minidisks. The source and the target device must be of the same type and must be in the same subsystem.

tcyl The decimal specification of where copying is to start on the target device. Cylinder is the smallest entity that can be specified for any SNAP command function. The target cyl specification (tcyl) added to the specified or calculated ncyl-1 value for the source device is the resulting target end cylinder address and it must not exceed the device's primary number of cylinders.

VOL1=volid
The VOL1 label that the TARGET device is to receive after the source volume has been copied. This operand is required if unique VOLIDs are to be maintained, otherwise the source and the target device would have the same VOL1 label after the copy function has completed. The VOL1 label specification for a target device will only be accepted when both, the cyl and the DSN= specification have been omitted (which means copying a full VOLUME).

NOCOPY
Indicates that a physical copy of the source data (Volume, DSN, or cylinders) on the specified target is not required. This keyword is useful when creating a backup tape and a physical copy is not required. When the backup tape has been created, the target device is usually no longer required. It can therefore be deleted (using DDSR) and the relation terminated. The NOCOPY relation exists until it is explicitly reset using the DDSR command.

However, you should be aware that the NOCOPY option does **not** imply that the target device will not be used. If the subsystem is running short of CACHE storage, it will use the TARGET device internally.

NOPROMPT
Prevents decision-type messages to be issued. Some messages require an operator reply before the specified function is going to be initiated. The specification of the NOPROMPT keyword will cause the system to bypass this decision-type message and will initiate the function without any additional notice.

DDSR Delete Data Space Request is a command requesting an eventual ongoing SNAP command to be terminated before physical copying of the entities specified in that SNAP command has been completed and/or the associated File-Id, if any, to be deleted from the VTOC. DDSR, if specified for a unit without any additional operands for a device which is the target device of a SNAP ...NOCOPY relation, will cause this NOCOPY relation to be terminated for the specified device. Since such a target device does not represent a physical copy of its associated source device, it must **NOT BE USED** once the DDSR command has been completed.

This requires the associated volume to be re-initialized (ICKDSF) before using it as a regular data-pack again (assuming it is not going to be used as a SNAP target device in which case no initialization is required). VSE requires the volume to be down (DVCDN command) if the whole volume is to be deleted.

unit This is the Device-ID (cuu) or the VOL1 label of the device that should either be totally released, or, in case a data-set-name (DSN=data-set-name) identifies the device containing the File-Id of the file those label information is to be deleted from the VTOC.

DSN= This is the data-set-name, identifying the file on the specified unit, which must be a non-VSAM file, that the operator wants to be deleted. If the specified unit is in the UP (DVCUP) state, then the label information for this file will be deleted from the VTOC. If the device is down (DVCDN), the command will be rejected and an error message is provided. Processing multi-volume-files is the responsibility of the operator, such that the DDSR command must be repeated for all the volumes containing file extents.

STATUS
The IXFP STATUS function provides information about the current status and progress of ongoing or persisting FlashCopy relations, for all ESS-subsystem devices. This function has no parameters. The ESS-subsystem devices must have been added during the IPL.

Using IXFP SNAP with VM Minidisks

You have to consider that for minidisks which are using MDC (Mini Disk Caching) the MDC buffer must be flushed before performing a SNAP or DDSR function, otherwise data can be incomplete. The MDC problem is solved by VM APAR VM61486.

Note: Other host caching products (for example, Cache Magic) will have the same requirements.

Issuing IXFP Commands From a Batch Job

A small REXX/VSE procedure can be used to issue IXFP commands from a batch job. The following example shows such a procedure:

```
* $$ JOB JNM=IXFPREXX,CLASS=0,DISP=D
// JOB IXFPREXX
// EXEC LIBR
ACC S=PRD2.CONFIG
CAT IXFPREXX.PROC  R=Y
/*    rexx/vse procedure     */
/* to issue console commands */
trace off
rc = SENDCMD('your-console-cmd-1')   /* enter your 1st IXFP cmd here */
call sleep 5                         /* wait for 5 seconds       */
rc = SENDCMD('your-console-cmd-2')   /* enter your 2nd IXFP cmd here */
exit rc
/+
/*
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE)
// EXEC REXX=IXFPREXX
/&
* $$ EOJ
```

More information on the REXX/VSE Console Automation Capability can be found in the *REXX/VSE Reference* manual, SC33-6642.

Using FlashCopy to Backup VSE/VSAM Datasets

Background

If you use FlashCopy to directly backup VSE/VSAM datasets, the following problems occur in the *snapped volume* (the target volume) that is produced by FlashCopy:

1. At DEFINE time of a VSAM dataset, all information about the dataset is saved in the catalog (dataset name, dataset attributes, VOLIDs, extent information, and so on).
2. Any access to a VSAM dataset is directed via:
   a. the name of its owning catalog
   b. the VOLID of the affected DASD volume, found in the catalog
   c. the name of the dataset, found in the catalog
   d. the extents of the dataset, found in the catalog
   e. certain dataset attributes (such as ″indexed″, for example).

Or in other words:

1. VSAM dataset names, VOLIDs, and extent information are saved in the catalog.
2. Access to a VSAM dataset requires the correct name of the catalog, the correct name of the dataset, and the correct VOLID.

In addition, duplicate VOLIDs cannot be used within the same z/VSE system, especially where VSE/VSAM datasets are accessed.

To overcome these limitations, the IDCAMS **SNAP** command must be used together with:
- IDCAMS BACKUP
- A synonym list

Run the following IDCAMS commands:

1. IDCAMS SNAP with a synonym list (similar to that used for the IDCAMS BACKUP) to control the source and snapped (target) volumes for the FlashCopy function. Snapped volumes are created. These snapped volumes are not available for normal VSAM access: they are therefore *very safe*.
   Please Note: The target device must be set UP (DVCUP command) prior to initiating the IDCAMS SNAP function. For the target, this is the *opposite* to what you would set for the IXFP SNAP command.
2. IDCAMS IMPORT CONNECT to inform the z/VSE system that a copy of the User Catalog now exists on the *snapped volume*, and that this copy of the User Catalog now has a *synonym name*.
3. IDCAMS BACKUP against the snapped volumes, using the synonym list.

Using the FlashCopy Dialog (Fastpath 3719)

The dialog shown in Figure 75 on page 293 provides a fast backup function from disk to **tape** for VSE/VSAM files. You can either:

- Copy a whole catalog and all its related files.
- Select individual files from the catalog for backup (IDCAMS BACKUP).

The job stream generated by the dialog includes three IDCAMS commands described previously (IDCAMS SNAP, IDCAMS IMPORT CONNECT, and IDCAMS BACKUP).

Note: A backup of snapped VSAM files requires the availability of the catalog and of **all** the related disk volumes holding data of the VSAM files owned by the catalog.

Figure 75 shows how you select the FlashCopy dialog.

```
IESADMSL.IESEVSAM        BACKUP/RESTORE VSAM OBJECTS
                                                    APPLID: DBDCCICS
  Enter the number of your selection and press the ENTER key:

        1  Export VSAM File
        2  Import VSAM File
        3  Backup VSAM File
        4  Restore VSAM File
        5  Export-Disconnect a User Catalog
        6  Import-Connect a User Catalog
        7  Copy In Catalog
        8  Copy Out Catalog
        9  FlashCopy VSAM Catalog/Files (ESS ONLY)






PF1=HELP                    3=END       4=RETURN                 6=ESCAPE(U)
                            9=Escape(m)

==>                                               Path: 371
```

Figure 75. Selecting the FlashCopy VSAM Catalog/Files Dialog

Figure 76 shows how you can use FlashCopy to specify a catalog to be backed up.

```
DSF$SNP1                 FLASHCOPY VSAM CATALOG/FILES

Enter the required data and press ENTER.


CATALOG TYPE................ 2          Enter  1 for MASTER catalog, or
                                         2 for USER catalog.

Enter  the  identification of the user catalog if you specify  2 for USER in
the CATALOG TYPE field.
      ________        ________        ________        ________        ________

Enter a synonym name for the snapped catalog.
      ________        ________        ________        ________        ________

BACKUP WHOLE CATALOG......... 1         Enter  1 to backup EVERYTHING in
                                        your catalog, or  2 to backup
                                        SELECTED files.
Only backup to tape is supported.

 PF1=HELP      2=REDISPLAY  3=END
```

Figure 76. FlashCopy VSAM Catalog/Files Dialog - Catalog Definitions

Figure 77 on page 294 shows how you can use FlashCopy to specify a source and target volume to be backed up.

```
 F$SNP2                    FLASHCOPY VSAM CATALOG/FILES

 Enter the required data and press ENTER.


Enter the Source Disk Volume where the CATALOG and all its Datasets reside
and the Target Volume to which the snapshot is to be done.

SOURCE VOLUME 1............. ______      Enter the Volume-id where the CATALOG
                                         resides

TARGET VOLUME 1............. ______      Enter the Volume-id to which Snap
                                         Shot has to be done

MORE VOLUMES................ 2           Enter 1 to add more volumes.
                                         Otherwise, enter  2




 PF1=HELP       2=REDISPLAY  3=END
```

Figure 77. FlashCopy VSAM Catalog/Files Dialog - Volume Definitions

Figure 78 shows how you can use FlashCopy to specify the characteristics of the backup tape.

```
 DSF$SNP3              FLASHCOPY VSAM CATALOG / FILES

 Enter the required data and press ENTER.


TAPE ADDRESS................ ___         Address of the tape unit (cuu).  For
                                         valid addresses enter a "?".
VOLUME SERIAL NUMBER......... ______     Volume serial number of the backup
                                         tape.
DENSITY/MODE................ __          Enter the density/mode. For valid
                                         densities/modes enter a "?".
RETENTION PERIOD............ ________    Enter the number of days (0-9999) or
                                         the Julian date in the form YY/DDD or
                                         YYYY/DDD.
REWIND OPTION............... 1           Enter 1 for REWIND,  2 for NOREWIND
                                         or  3 for UNLOAD.
LABEL PROCESS............... _           Enter 1 if you want a labeled
                                         tape, otherwise enter 2.


 PF1=HELP       2=REDISPLAY  3=END
```

Figure 78. FlashCopy VSAM Catalog/Files Dialog - Backup Tape Characteristics

Advantages in Creating a Snapshot of Entire Disk Volumes

These are the advantages in using IDCAMS SNAP together with a synonym list, to produce “snapshots” of volumes:

1. After a snapshot of disk volumes is complete, backup processing *can be started immediately* and can run *during online processing*.
2. During the very short “copy-time” of the snapshot (which may be minutes, or even **seconds**), online systems only have to be shut down for a very short time.
3. There is no risk of losing data.
4. There is faster access to the snapped (target) datasets.
5. There are no catalog changes on the snapped (target) catalog.
6. Error-prone and time-consuming catalog changes are not required on the snapped catalog. The snapped catalogs remains 100% unchanged. The snapped

catalogs and the datasets could then be used for disaster recovery (for example, to replace one or more complete disk volumes).

7. When the snapshot is made, the contents of the snapped (target) volumes remain identical to the contents of the source volumes. This is very important if you must carry out a disaster recovery.
8. Any BACKUP jobs that you run after the snapshot, will use “frozen” data on the snapped volumes. This data is independent of any data changes that take place on the online systems.
9. Running IDCAMS SNAP and IDCAMS BACKUP using a synonym list significantly reduces the time during which your system is unavailable.

Using IDCAMS SNAP and BACKUP With a Synonym List

These are the steps you should follow:

1. **Run IDCAMS SNAP using the Synonym List.**

   Use the IDCAMS SNAP to create a snapshot of all entire disk volumes where the catalog and all its datasets reside. Give the snapped (target) volumes different VOLIDs than the source volumes.
   Note: The snapped (target) volumes will then be online after the IDCAMS SNAP has run.

   This is the syntax of the IDCAMS SNAP command:

   ```
   SNAP
         SOURCEVOLUMES(volser[ volser...])
         TARGETVOLUMES(volser[ volser...])
         [NOPROMPT|PROMPT]
   ```

   where:

   SOURCEVOLUMES(volser[volser...])
   TARGETVOLUMES(volser[volser...])
   Are a pair of lists indicating from which volumes, and to which volumes, the snapshot is to be done.
   Please Note: The target device must be set UP (DVCUP command) prior to initiating the IDCAMS SNAP function. For the target, this is the *opposite* to what you would set for the IXFP SNAP command.
   Abbreviations: SVOLUME or SVOL, TVOLUME or TVOL

   NOPROMPT
   This keyword prevents decision-type messages from being issued.

   PROMPT
   This keyword allows decision-type messages to be issued.

2. **Run IDCAMS IMPORT CONNECT.**

   You run IDCAMS IMPORT CONNECT to inform the z/VSE system that a copy of the User Catalog now exists on the *snapped (target) volume*, and that this copy of the User Catalog now has a *synonym name*.

   Notes:

   a. The catalog that has a synonym name now exists (including its datasets), but *cannot be accessed* by normal applications.
   b. You use a snapped (target) Master Catalog in the same way as you use a snapped (target) User Catalog.

3. **Run IDCAMS BACKUP using the Synonym List.**

   IDCAMS BACKUP uses the parameters contained in the synonym list:

   - The synonym list is used to route the VSAM OPEN and BACKUP functions to the snapped (target) volumes.

- The BACKUP works in the same way as previously, *except* that it uses the synonym list to access the snapped volumes. Therefore, you can use all features of IDCAMS BACKUP.
- You can use the output from this IDCAMS BACKUP (that uses a synonym list) in the same way as before (for IDCAMS RESTORE).

This is the syntax of the IDCAMS BACKUP command:

```
BACKUP  ..........
        SYNONYMLIST(                          -
        SOURCEVOLUMES(volser[ volser...]) -
        TARGETVOLUMES(volser[ volser...]) -
        CATALOG(catname[/password])       -
        SYNONYMCATALOG(catname[/password]) )
```

where:

SYNONYMLIST
Indicates that this backup uses a synonym list of ″snapped″ VSAM volumes.
Abbreviations: SYNLIST or SYNL

SOURCEVOLUMES(volser[volser...])
TARGETVOLUMES(volser[volser...])
Are a pair of lists indicating from which volumes, and to which volumes, snapshot has to be done.
Abbreviations: SVOLUME or SVOL, TVOLUME or TVOL

CATALOG(catname[/password])
Specifies the name and the password of the source catalog, which is the original catalog from which a snapshot was done.
Abbreviation: CAT

SYNONYMCATALOG(catname[/password])
Specifies the synonym name and password of the snapped (target) catalog which was copied (using snapshot) to the snapped (target) volume. You must ensure that the synonym name of the catalog has been imported using IMPORT CONNECT, before running the IDCAMS BACKUP. The password is identical to the password of the source catalog. **Note:** You must provide the synonym name of the catalog in a DLBL statement containing filename IJSYSUC.
Abbreviation: SYNCAT

For reference purposes, Table 8 shows the internal structure of the synonym list:

Table 8. Internal Structure of Synonym List

Offset	Source	Target
0	Header with Eye Catcher	
8	Source Catalog / Source Password	Target Catalog / Target Password
112	Source Volume 1	Target Volume 1
124	Source Volume 2	Target Volume 2
...	...	...
	End of List	

Example of Running IDCAMS SNAP /BACKUP With a Synonym List

The following jobstream example shows how to:

1. Create a snapshot of the *source* volumes *SYSWK1* and *DOSRES* to the *snapped (target)* volumes *VOLSN1* and *VOLSN2*.
2. Run an IDCAMS IMPORT CONNECT to inform the z/VSE system that a copy of the User Catalog (VSESP.USER.CATALOG on SYSWK1 and DOSRES) now exists on the snapped (target) volume, and that this copy of the User Catalog now has a *synonym name* (VSESP.SNAP.CATALOG).
3. Run an IDCAMS BACKUP that uses the parameters contained in the synonym list.

Input

```
// JOB SNAP and BACKUP from snapped Volumes
// ASSGN SYS005,180
// DLBL IJSYSUC,'VSESP.SNAP.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 /* First: do the SNAPSHOT */       -
  SNAP
     SOURCEVOLUMES(SYSWK1 DOSRES)  -
     TARGETVOLUMES(VOLSN1 VOLSN2)
 /* Second: Synonym Name for the snapped Catalog */-
  IMPORT CONNECT OBJECTS((VSESP.SNAP.CATALOG       -
     VOLUMES(VOLSN1) DEVT(3390)))                  -
     CATALOG(VSAM.MASTER.CATALOG)
 /* Third: Backup from snapped volumes */-
  BACKUP (*)                            -
     SYNONYMLIST(                       -
     SOURCEVOLUMES(SYSWK1 DOSRES)       -
     TARGETVOLUMES(VOLSN1 VOLSN2)      -
     CATALOG(VSESP.USER.CATALOG)        -
     SYNCATALOG(VSESP.SNAP.CATALOG)     )
/*
/&
```

You can also use the dialog *Flashcopy VSAM Catalog/Files* (Fastpath 3719) to create such a job stream. For details, refer to Chapter 11, “Managing VSE/VSAM Files and Catalogs,” on page 241.

Output

The output from using IDCAMS BACKUP (with a synonym list) is any normal backup media (tape or disk). This is the same as any other IDCAMS BACKUP output.

Using IXFP SNAP Function With VM Minidisks

The IXFP SNAP function, if used on a z/VSE system running under VM, does not allow volume or cylinder relocation for **partial minidisks**.

You have to consider that for minidisks which are using MDC (Mini Disk Caching) the MDC buffer must be flushed before performing a SNAP or DDSR function, otherwise data can be incomplete. The MDC problem is solved by VM APAR VM61486, which is included in z/VM Version 4 onwards.

Note: Other host caching products (for example, Cache Magic) will have the same requirements.

VSE/Fast Copy (FCOPY) Exploitation of FlashCopy

The *VSE/Fast Copy* utility (described in the manual *z/VSE System Utilities*) exploits the FlashCopy function. It supports **full volume** backup from **disk to disk** but not the copying of files or extents.

The following VSE/Fast Copy commands are supported:
- COPY ALL
- COPY VOLUME

If FlashCopy support is available in the disk hardware, it will be used for which a full volume backup request was issued. If the requested support is not available, a normal VSE/Fast Copy backup will be performed.

The following VSE/Fast Copy optional parameters are supported:

```
IV  (input volume)
OV  (output volume)
NV  (new volume)
```

The following VSE/Fast Copy optional parameters are tolerated:

```
NOPROMPT
NOVERIFY
LIST
```

The following VSE/Fast Copy optional parameters force FlashCopy NOT to be used. These parameters apply to COPY VOLUME only:

```
EXCLUDE
NOVSAM
NOEXPIRED
```

Job Stream Example

The following job stream example applies to the COPY ALL as well as to the COPY VOLUME command:

```
// JOB jobname
// ASSGN SYS004,140
// ASSGN SYS005,141
// EXEC FCOPY
COPY VOLUME IV=SYSRES
/*
/&
```

Chapter 16. Supporting Application Development

This chapter describes two administrative tasks for application development:
- Tailoring compile skeletons.
- Creating application job streams via dialog.

Tailoring Compile Skeletons

"Handling VSE/ICCF Library Members" in the manual *VSE/ESA Programming and Workstation Guide* describes the *Program Development Library* dialog. You use this dialog to access and work with VSE/ICCF libraries. Various options allow you to create, maintain, and process library members. One of the options allows you to compile library members.

Before using the *Compile a Member* option, the related compile skeletons must be tailored as needed for your installation.

The compile skeletons are available in VSE/ICCF library 2. The skeleton names are

C$$xxyyy and **C$Qxxyyy**

where Q identifies the skeletons provided for use with the DB2 Server. For a complete list of the skeletons available refer to the manual *z/VSE Planning* under "Tailoring Compile Skeletons".

xx can be:
CN (for C for z/VSE)
CV (for COBOL for z/VSE)
PV (for PL/I for z/VSE)
AS (for High Level Assembler for VSE)
RP (for RPG II)
FO (for VS FORTRAN)

RPG II support is only available for CICS/VSE 2.3 online programs (ONL).

yyy can be:
ONL for online program.
BAT for batch program.
SUB for batch subroutine.
MAP for BMS map definition. **Note:** You can also use the TEMPLATE parameter of the *Compile Job Generation* panel (Fastpath 51) to generate an HTML map definition from a BMS MAP (for details, refer to the *VSE/ESA Programming and Workstation Guide*).

Before you tailor the skeletons, you should consider who will use them and how they will be used.
1. You can give the skeletons to all or some application programmers. Copy them from library 2 to a library to which the programmer has write access. The programmer can then tailor the skeletons.
2. You can tailor the skeletons for the entire system. In this way, you can establish certain standards for compile jobs and have every programmer use the same skeletons. For this method, tailor the skeleton files and leave them in library 2.

When a user selects the COMPILE option in the *Program Development Library* dialog, the system searches for the compile skeleton in the following order:
1. User's primary library
2. User's current secondary library (if any)
3. Common library (VSE/ICCF library 2)

   Note: From z/VSE 3.1 onwards, HLASM no longer uses work files. You must therefore ensure that your partition storage is accordingly large.

Example: Skeletons C$$ASBAT and C$$ASONL

Figure 79 shows skeleton C$$ASBAT for compiling a High Level Assembler **batch** program. Figure 80 on page 302 shows skeleton C$$ASONL for compiling a High Level Assembler **online** (CICS) program.

You should tailor the skeletons according to the needs of your program development environment. Use the information provided for the two examples when tailoring other skeletons.

Notes:

1. In the skeletons and compile examples on the following pages the statement

   ```
   // EXEC ASMA90,SIZE=(ASMA90,64K)....
   ```

   calls the High Level Assembler. Note that the 'C' at the end of the first line is the continuation character. Refer to the manual *z/VSE Planning* under "Changing from DOS/VSE Assembler to High Level Assembler" for further details about the High Level Assembler.
2. Instead of using work files, the High Level Assembler uses partition GETVIS storage. You must therefore ensure that the partition you are using has sufficient GETVIS storage.

Skeleton C$$ASBAT

```
* $$ JOB JNM=&JOBNAME,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB &JOBNAME COMPILE PROGRAM &PROGNAME
// SETPARM CATALOG=&CATALOG
// IF CATALOG = 2 THEN
// GOTO NOCAT
// LIBDEF PHASE,CATALOG=lib.sublib
// OPTION ERRS,SXREF,SYM,NODECK,CATAL
   PHASE &PROGNAME,*
// GOTO ENDCAT
/. NOCAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
/. ENDCAT
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
               -200K,ABOVE)'
* $$ SLI ICCF=(&PROGNAME,&PASSWORD),LIB=(&LIBNO)
/*
// IF CATALOG EQ 2 OR $MRC GT 4 THEN
// GOTO NOLNK
// EXEC LNKEDT,SIZE=256K
/. NOLNK
/&
* $$ EOJ
```

Figure 79. Compile Skeleton (C$$ASBAT) for Batch High Level Assembler Programs

Batch and Online Skeleton Information

The following information applies in general to any batch and online compile skeleton. Note that parameters beginning with an & are replaced with information the user enters when using the COMPILE option. You must change the LIBDEF statement and replace the variable **lib.sublib** with the library and sublibrary name you want to use. In addition, review and consider changing the following:

- Job class and disposition in the * $$ JOB statement.

  ```
  DISP=D
  CLASS=A
  ```

- Print class and disposition in the * $$ LST statement.

  ```
  DISP=D
  CLASS=Q
  ```

- Additional LIBDEF statements for your own libraries should be added to the supplied LIBDEF statements and inserted after each job statement.
- Check all C/370 skeletons whether the DLBL for the C/370 message file is the same you used during the installation of C/370.
- COBOL options in the OPTION statements.
- For Map Definitions (BMS map) it is possible to also generate HTML templates. These templates are stored in PRD2.DFHDOC and are used with the CICS Web Support (CWS).
- CICS preprocessor options (online program).
- SLI statement.

  VSE/POWER and VSE/ICCF features allow you to include a VSE/ICCF member at **execution** time. This reduces submit time and uses disk space more efficiently. The member does not have to be transferred from the VSE/ICCF library to the VSE/POWER reader queue. Consider the following about these features:

  1. You should **not** change the VSE/ICCF member until the job completes **or** if you need to change it, replace the SLI statement with a **/INCLUDE** statement. This will put the member in the reader queue at submit time.
  2. If the compile job stream runs on another system, a correctly named VSE/ICCF member must be available at the remote system **or** if you want a member at your system compiled at another system, replace the SLI statement with a **/INCLUDE** statement. An incorrect VSE/ICCF member may cause unpredictable results during execution.

Note that names of CICS tables must start with **DFH**. When assembling CICS tables and this naming convention is not observed, the job stream does not work correctly.

Do **not** change the * $$ PUN and $ $$ PUN statements, otherwise program IESINSRT does not work correctly.

Program IESINSRT

Program IESINSRT supports any nesting level and copies all input from SYSIPT to SYSPCH up to the statement * $$ END. Statement * $$ END itself is not copied, but causes the program to exit. The punched output of IESINSRT is used to build a new entry in the VSE/POWER reader queue since the VSE/POWER JECL statement * $$ PUN DISP=I,PRI=9,CLASS=A is part of the JCL for IESINSRT. Statement * $$ END may be hidden like a VSE/POWER JECL statement by a $ instead of an *. On the highest nesting level:

- All statements starting with $ $$ are changed to * $$.
- Statements /* and /& may be hidden by a # instead of a /.

- All statements starting with # are changed to /. The # is used to avoid a premature EOF (end-of-file) condition.
- All // JOB and /& statements must be hidden by a #. The reason is that all JCL statements between a GOTO and the target label statement are ignored except for // JOB and /&. If these statements are not hidden by a #, they cause a job termination.
- If a complete job is generated by IESINSRT, make sure you also generate VSE/POWER JECL statements. Otherwise, VSE/POWER generates an AUTONAME job.

Skeleton C$$ASONL

```
----------------------------------------------------- JOB 1 (Part 1) -------------
* $$ JOB JNM=&JOBNAME,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
* $$ PUN DISP=I,PRI=9,CLASS=A
// JOB &JOBNAME TRANSLATE PROGRAM &PROGNAME
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
------------------------------------------------------ JOB 1 (Data Part 1) ---------
$ $$ LST DISP=D,CLASS=Q,PRI=3
// JOB &JOBNAME COMPILE PROGRAM &PROGNAME
// SETPARM CATALOG=&CATALOG
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
// GOTO ENDCAT
------------------------------------------------------- JOB 2 (Part 1) --------------
/. CAT
// LIBDEF PHASE,CATALOG=lib.sublib
// OPTION ERRS,SXREF,SYM,CATAL,NODECK
   PHASE &PROGNAME,*
   INCLUDE DFHEAI
/. ENDCAT
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
               -200K,ABOVE)'
* $$ END
------------------------------------------------- JOB 1 (Part 2) -----------
// ON $CANCEL OR $ABEND GOTO ENDJ2              (this job generates:
// OPTION NOLIST,NODUMP,DECK                     Part 2 of JOB 2)
// EXEC DFHEAP1$,SIZE=512K
*ASM XOPTS(CICS)
* $$ SLI ICCF=(&PROGNAME,&PASSWORD),(LIB=&LIBNO)
/*
/. ENDJ2
// EXEC IESINSRT
/*
------------------------------------------------- JOB 1 (Data Part 2) --------
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO NOLNK
// EXEC LNKEDT,SIZE=256K
------------------------------------------------- JOB 2 (Part 3) -------------
/. NOLNK
#&
$ $$ EOJ
------------------------------------------------- JOB 1 (Part 3) --------------
* $$ END
/&
* $$ EOJ
```

Figure 80. Compile Skeleton (C$$ASONL) for Online High Level Assembler Programs

Note that JOB 1 (Part 2) contains the statement:

```
// EXEC DFHEAP1$,SIZE=512K
```

The CICS preprocessor for High Level Assembler programs punches the preprocessed source code as Part 2 of JOB 2. Data Part 2 of JOB 1 is punched as Part 3 of JOB 2 by IESINSRT.

Compile Example (Part 1)

Skeleton C$QASONL is used as an example for showing a compile skeleton (Figure 81) and the jobs it generates (Figure 82 on page 304).

If the **DB2 Server for VSE** is not installed in **PRD2.DB2740**, you must change the LIBDEF statement accordingly.

```
-------------------------------------------- JOB 1 (Part 1)-----------------------
* $$ JOB JNM=&JOBNAME,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
* $$ PUN DISP=I,DEST=*,PRI=9,CLASS=A
// JOB &JOBNAME DB2 PRE PROCESS &PROGNAME
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
-------------------------------------------- JOB 2 (Part 1) ----------------------
$ $$ LST DISP=D,CLASS=Q,PRI=3
$ $$ PUN DISP=I,DEST=*,PRI=9,CLASS=A
// JOB &JOBNAME CICS PRE PROCESS &PROGNAME
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
-------------------------------------------- JOB 3 (Part 1) ----------------------
$ $$ LST DISP=D,CLASS=Q,PRI=3
// JOB &JOBNAME COMPILE PROGRAM &PROGNAME
// LIBDEF *,SEARCH=PRD2.DB2740
// SETPARM CATALOG=&CATALOG
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
// GOTO ENDCAT
/. CAT
// LIBDEF PHASE,CATALOG=lib.sublib
// OPTION ERRS,SXREF,SYM,CATAL,NODECK
   PHASE &PROGNAME,*
   INCLUDE DFHEAI
/. ENDCAT
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
               -200K,ABOVE)'
$ $$ END
----------------------------------------------- JOB 2 (Part 2) -------------------
// ON $CANCEL OR $ABEND GOTO ENDJ3
// OPTION NOLIST,NODUMP,DECK
// EXEC DFHEAP1$,SIZE=512K
*ASM XOPTS(CICS)
* $$ END
----------------------------------------------- JOB 1 (Part 2) -------------------
// ON $CANCEL OR $ABEND GOTO ENDJ2
// LIBDEF *,SEARCH=PRD2.DB2740
// EXEC PROC=ARIS74DB
// EXEC PROC=ARIS74PL
// EXEC ARIPRPA,SIZE=AUTO,PARM='ISOLATION(&ISOL),&BLOCK,DBNAME=&DBNA,
               USERID=&SQLUSERID/&SQLPW,PREP=&PROGNAME'
* $$ SLI ICCF=(&PROGNAME,&PASSWORD),LIB=(&LIBNO)
/*
/* ENDJ2
// EXEC IESINSRT
----------------------------------------------- JOB 2 (Part 3) -------------------/*
/*
/. ENDJ3
```

Figure 81. Compile Skeleton (C$QASONL) for Online High Level Assembler Programs for DB2 (Part 1 of 2)

```
// EXEC IESINSRT
---------------------------------------------- JOB 3 (Part 2) --------------------
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO NOLNK
   INCLUDE ARIRRTED
// EXEC LNKEDT,SIZE=256K
/. NOLNK
#&
$ $$ EOJ
$ $$ END
------------------------------------------------- JOB 2 (Part 4) -----------------
#&
$ $$ EOJ
* $$ END
---------------------------------------------- JOB 1 (Part 3) --------------------
/&
* $$ EOJ
```

Figure 81. Compile Skeleton (C$QASONL) for Online High Level Assembler Programs for DB2 (Part 2 of 2)

Compile Example (Part 2)

The following 3 jobs are generated by skeleton C$QASONL (shown in Figure 81 on page 303).

```
--------------------------------------------- JOB 1 ------------------------------
* $$ JOB JNM=COMUSER,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
* $$ PUN DISP=I,PRI=9,CLASS=A
// JOB COMUSER DB2 PRE PROCESS TEST
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
// ON $CANCEL OR $ABEND GOTO ENDJ2
// LIBDEF *,SEARCH=PRD2.DB2740
// EXEC PROC=ARIS74DB
// EXEC PROC=ARIS74PL
// EXEC ARIPRPA,SIZE=AUTO,PARM='ISOLATION(CS),NOBLK,                    *
               USERID=SQLDBA/SQLDBAPW,PREP=TEST'
* $$ SLI ICCF=(TEST),LIB=(0099)
/*
/. ENDJ2
// EXEC IESINSRT
/&
* $$ EOJ
------------------------------------------ JOB 2 ---------------------------
* $$ LST DISP=D,CLASS=Q,PRI=3
* $$ PUN DISP=I,PRI=9,CLASS=A
// JOB COMUSER CICS PRE PROCESS TEST
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
```

Figure 82. Jobs Generated by Compile Skeleton C$QASONL (Part 1 of 2)

```
// ON $CANCEL OR $ABEND GOTO ENDJ3
// OPTION NOLIST,NODUMP,DECK
// EXEC DFHEAP1$,SIZE=512K
*ASM XOPTS(CICS)
/*
/. ENDJ3
// EXEC IESINSRT
/&
* $$ EOJ -------------------------------------------- JOB 3 -----------------------------
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB COMUSER COMPILE PROGRAM TEST
// SETPARM CATALOG=2
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
// GOTO ENDCAT
/. CAT
// LIBDEF PHASE,CATALOG=lib.sublib
// LIBDEF *,SEARCH=PRD2.DB2740
// OPTION ERRS,SXREF,SYM,CATAL,NODECK
   PHASE TEST,*
   INCLUDE DFHEAI
/. ENDCAT
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
               -200K,ABOVE)'
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO NOLNK
   INCLUDE ARIRRTED
// EXEC LNKEDT,SIZE=256K
/. NOLNK
/&
* $$ EOJ
```

Figure 82. Jobs Generated by Compile Skeleton C$QASONL (Part 2 of 2)

For EXEC CICS batch client programs, skeletons are available in VSE/ICCF library 59 as follows:

SKEXCIAS High Level Assembler for VSE
SKEXCICV COBOL for VSE/ESA
SKEXCIPL PL/I for VSE/ESA
SKEXCICN C for VSE/ESA

Creating an Application Job Stream

The *Create Application Jobstream* dialog helps you create job streams. You can save the input parameters that you specify in a VSE/ICCF library member for future use. If you create another job stream with similar parameters, you can use the saved input for default values.

To access the dialog, start with the *z/VSE Function Selection* panel and select:

5 (Program Development)
2 (Create Application Job Stream)

Administrator Fast Path: 52	Synonym Default: ____ Yours:

The programmer (PROG) must choose selection **3** on the *z/VSE Function Selection* panel. The dialog displays the *Create Application Job Stream: Create or Modify* panel.

If you used the dialog before and saved your input parameters, you can use them as defaults. This is helpful if you are creating a job stream with parameters that are similar to ones in a previous job stream. The input was saved in a VSE/ICCF library member. Specify the name of the member on the panel.

If you are creating a new job stream, simply press **ENTER**. You are asked to define new parameters. On the next panel, specify the following:

PROGRAM NAME
Enter the name of the program. The name cannot be:
- ALL
- ROOT
- S

LIBRARY and SUBLIBRARY NAME
Specify the library and sublibrary where the program resides. The dialog searches for the program in this library and sublibrary.

If the sublibrary is defined in the LIBDEF PHASE,SEARCH= statement for the partition where the job runs, you can leave these fields blank.

From the *Select Functions* panel, select any optional functions for your job stream. If your program uses any Input/Output (I/O) devices, other than disk, you can provide specifications for individual devices. Enter:
1 - YES (Provide specifications)
2 - NO (Do not provide specifications)

If you specify 1 (YES), the dialog displays additional panels. The options you can specify are described in the following sections, beginning with "Printer Specifications" on page 307.

When you finish entering your options (or choose not to specify any), you can do one of two things:
1. Return to the beginning of the dialog to review and possibly change the options.

   The dialog redisplays the *Select Functions* panel. Again, indicate the options you want to review or specify.
2. Continue with the dialog.

After you continue, you are asked whether you want to save the parameters. If you want to save them, enter the name of a VSE/ICCF library member. The dialog stores the parameters in the member in your default primary library.

If you do not want to save the values, just press ENTER.

The dialog creates a job with the default name PRGEXE. On the *Job Disposition* panel, you can submit the job to batch, file it in your default primary library, or both.

Printer Specifications

You can define the following requirements for up to three printouts:

- Printer address
- Logical unit

  Specify **SYSLST** or **SYS000 - SYS254**.[1]
- Output class
- Number of copies
- Form number

  This specifies that a special form is used for the output.
- Forms control buffer (FCB)
- Train image buffer (UCB)

Reader or Punch Specifications

You can specify the following units to be used by your program:

- Reader logical unit

  If your program has card input from a reader other than SYSRDR, specify the logical unit (**SYS000 - SYS254**).[1]
- Punch or punch to tape
  - Logical unit

    If your program does not write to SYSPCH, specify the logical unit (**SYS000 - SYS254**).
  - Tape address

    If your program punches to tape, enter the physical tape address.

Tape Specifications

You can define the following tape I/O specifications for up to four tapes:

- Tape address
- Logical unit

  Specify the logical unit which your program uses to reference the tape (**SYS000 - SYS254**).[1]
- Tape volume ID

  If you specify an ID, write down the value you use. You will need to know it later, if you use the tape for input.

  You should have comments and a PAUSE statement in the job stream for tape mount instructions.
- File name (name which your program uses to reference the tape file).
- File ID

  Specify an optional name that is associated with the file on the tape. If you enter a file ID, write down the value you specify. You will need to know it later when you process the file.
- File date

  For output tapes, this is the expiration date. For input tapes, it is the creation date.

  The date format is **YYYY/DDD**, where YYYY is the year and DDD is the day of the year.

---

1. Please take account of the NPGR value that is set in the ALLOC procedure. This value might be lower than 255.

Diskette Specifications

You can define the following diskette I/O specifications:

- Logical unit

  Specify the logical unit which your program uses to reference the diskette (**SYS000 - SYS254**).[2]
- Volume identifier

  This is used to check that the correct volume is mounted.
- File name (name which your program uses to reference the diskette file).
- File ID
- Expiration date

  For output files, this is the date through which the file is kept. This field is not used for input files.

  The date format is **YY/DDD**, where YY is the year and DDD is the day of the year.

Data Specifications

You can choose how data is included in your job stream:

1. From a VSE/ICCF library member when the job stream runs.

   You are asked for the name of the library member that contains the data. Specify the password, if the member is password-protected.
2. Data entered from the dialog.

   You can enter up to three lines of data. The dialog includes the data in the job stream.

Job Information Specifications

You can specify the following job options:

- UPSI

  You can set up to eight user program switches. Positions 0 - 7 of the UPSI byte are set from left to right. For each program switch, specify:
  0 - Switch is set off
  1 - Switch is set on
  x - Switch is unchanged
- Job date

  Specify a date to override the system date.

In addition, you may include COMMENT and PAUSE statements.

2. Please take account of the NPGR value that is set in the ALLOC procedure. This value might be lower than 255.

Chapter 17. Maintaining and Cataloging Printer Information

Maintaining Printer FCB

You use the *Maintain Printer FCB* dialog to define and maintain printer forms control buffers (FCBs). For the FCBs provided by z/VSE, refer to "Creating Print Buffers for a System Printer" in the manual *z/VSE Installation*.

To access the dialog, start with the *z/VSE Function Selection* panel and select:

2 (Resource Definition)
4 (Hardware Configuration and IPL)
4 (Maintain Printer FCB)

Administrator Fast Path: 244	Synonym Default: fcb Yours:

The panel you get displays the FCBs currently defined. You can scroll through the entries using **PF7** and **PF8**. The options you can select are at the top of the display. Enter the option number in the OPT column to the left of the FCB name you want to process.

When you are finished, press **PF5**. The dialog processes the information and redisplays the selection panel.

The options you can select for maintaining FCBs are:

1 (Add)
2 (Alter)
5 (Delete)
7 (Catalog)

The dialog creates a job for cataloging the new or changed FCB in the IJSYSRS.SYSLIB library. After pressing **PF5** you get the *Job Disposition* panel. With it, you can submit the job to batch, file it in your default VSE/ICCF primary library, or both.

Add or Change an FCB

If you add (ADD) or change (ALTER) an FCB, you need the following information on the printer's characteristics:

FCB NAME
If you add an FCB, enter the new FCB name. If you alter an FCB, you can change the name.

The name identifies the FCB to the Interactive Interface. It is **not** the phase name. You can use the same name that you use for the phase name. The FCB name must be unique.

DEVICE TYPE
Enter **3203, 3211, 3262, 3289, 4245, 4248**, or **6262**.

PHASE NAME
The library phase name. You use this name either in the LFCB command or in a VSE/POWER LST control statement to load the FCB.

The system automatically loads one FCB at IPL. The phase name is fixed for each printer type. Unless an FCB has the standard name for the printer type, it must be loaded by the operator or by VSE/POWER JECL.

LINES PER INCH
Enter either **6** or **8** for the number of lines per inch. If you want to change the lines-per-inch setting anywhere on the form (when using an IBM 4248 or IBM 3262 printer for example), leave this field blank.

FORM LENGTH
Enter the page length, in inches. For example; 11, 12, and 8.5 are acceptable.

3211 INDEXING
Specify the following information for IBM 3211 printers with the indexing feature:

- SHIFT DIRECTION
  0 - No indexing
  1 - Right
  2 - Left
- SHIFT NUMBER

  The number of positions to be shifted (**1 - 31**). For no indexing, enter **0**.

CHANNEL POSITIONS
Enter the print line position for channels 1 - 12. If the channel is not used, enter **0**.

Channel 1 **cannot** be 0.

VERIFICATION MESSAGE
Enter a message that is printed on SYSLST when the FCB is loaded. The message is printed after the following header (where xxxxxxxx is the phase name):

```
xxxxxxxx LOADED
```

The following parameters are only required for **IBM 4248** printers.

STACKER-LEVEL CONTROL
Specify **1**, **2**, **3**, or **4**. The meaning is as follows:

```
1 = Automatic stacker level control.
2 = 25 mm (1 in) below automatic stacker level control.
3 = 51 mm (2 in) below automatic stacker level control.
4 = 76 mm (3 in) below automatic stacker level control.
```

HORIZONTAL COPY
Specify **1** (for YES) or **2** (for NO).

PRINT SPEED
Specify **1**, **2**, **3**, or **4**. The meaning is as follows:

```
1 = No change of the print-speed setting.
2 = Low speed    (2.200 lines per minute).
3 = Medium speed (3.000 lines per minute).
4 = High speed   (3.600 lines per minute).
```

TRAY DROP RATE
Specify **1**, **2**, **3**, or **4**. The meaning is as follows:

1 = 7 forms cause a drop.
Form thickness: 0.5 mm (0.02 in).
2 = 13 forms cause a drop.
Form thickness: 0.2 mm (0.007 in).
3 = 19 forms cause a drop.
Form thickness: 0.1 mm (0.005 in).
4 = 25 forms cause a drop.
Form thickness: < 0.1 mm (0.003 in).

OFFSET COUNT BYTE
If *horizontal copy* is on, the offset count byte specifies where the duplicate print line begins.

The dialog creates a job for cataloging the new or changed FCB in the IJSYSRS.SYSLIB library. After pressing **PF5** you get the *Job Disposition* panel. With it, you can submit the job to batch, file it in your default VSE/ICCF primary library, or both.

Additional Considerations

If you want to use an FCB that you create, you must load it into the printer. No re-IPL is necessary. The operator (or VSE/POWER JECL) can load it by using the LFCB command.

Cataloging Printer UCB

You use the *Catalog Printer UCB* dialog to catalog a universal character set buffer (UCB). For the UCBs provided by z/VSE, refer to the manual *z/VSE Installation* under "Creating Print Buffers for a System Printer".

A UCB converts bit patterns sent to the printer into specific locations on the print train. By using a UCB, you can take advantage of options such as different print trains, and upper- and lowercase printing.

The *Catalog Printer UCB* dialog creates a job to catalog a standard UCB or to assemble and catalog a non-standard UCB. To access this dialog, start with the initial *z/VSE Function Selection* panel and select:

2 (Resource Definition)
4 (Hardware Configuration and IPL)
5 (Catalog Printer UCB)

```
Administrator
Fast Path:  245
```

```
Synonym
Default:      ucb   Yours:
```

You get the *Catalog Printer UCB: Select Function* panel. From this panel you can select three different catalog dialogs:

1 (Catalog IPL loaded standard train)

Select this option to define a new UCB that is automatically loaded at IPL. The new UCB replaces the current UCB that is loaded at IPL.

2 (Catalog user loaded standard train)

Select this option if you want to define a new UCB, but you do **not** want to replace the current UCB loaded at IPL. The new UCB is assigned a name that can be used when a program is run.

3 (Catalog user loaded non-standard train)

Select this option if you want to define your own load phase. The source for the phase is in a VSE/ICCF library member. The dialog helps you assemble and catalog the non-standard UCB. The current UCB is still loaded at IPL.

After you make your selection, the dialog displays various panels. The input you need depends on the type of UCB you catalog (standard or non-standard). Review the descriptions below.

Standard UCB

If you select option 1 or 2 (standard train), you need the following information:

PRINTER TYPE
Select the printer type you are using.

PRINT TRAIN
Select the train type for the printer you are using.

BUFFER NAME
This is the phase name that is assigned to the UCB in the VSE library. The name **cannot** be:
- ALL
- S
- ROOT

You are asked for the buffer name, only if you are cataloging a user loaded standard train (option 2 on the selection panel).

After entering the information required, the *Job Disposition* panel is displayed. With it, you can submit the job to batch, file it in your VSE/ICCF primary library, or both.

Non-Standard UCB

If you select option 3 (non-standard train), you need the following information:

MEMBER NAME
The name of the VSE/ICCF library member that contains the source for the UCB you create.

PASSWORD
The password for the VSE/ICCF member. This is needed, if the member is password-protected.

LIBRARY NUMBER
The number of the library that contains the VSE/ICCF member.

BUFFER NAME
The phase name that is assigned to the UCB in the VSE library. The name **cannot** be:
- ALL
- S
- ROOT

After entering the information required, the *Job Disposition* panel is displayed. With it, you can submit the job to batch, file it in your VSE/ICCF primary library, or both.

Additional Considerations

1. If you select option 1 (Catalog IPL Loaded Standard Train), the dialog catalogs the UCB in the system library with a standard name. The system automatically loads it into the printer at IPL.
2. If you select option 2 (Catalog User Loaded Standard Train), you define the UCB with a phase name. The operator has to load it into the printer (with the LUCB command) before you can use it.
3. If you select option 3 (Catalog User Loaded Non-Standard Train), you must create a VSE/ICCF library member that contains the source for the UCB phase. The operator has to load the UCB into the printer (with the LUCB command).

Cataloging Your Own Print Control Buffer Phases

This step applies if the print trains (or belts) used on your location's printers do not match the default FCB and UCB images that are loaded automatically during IPL. You may catalog your own FCB (forms control buffer) or UCB (universal character set buffer) image phases, if there is a need. For a list of default FCB and UCB image phases, refer to the manual *z/VSE System Control Statements* under "Buffer Load Phases"; this list includes the names of the UCB-image object modules shipped with z/VSE for possible link-editing. The manual describes, in addition, how to create FCB or UCB image phases and how to catalog them. Your control statements for cataloging buffer-image phases (assuming that an IBM supplied UCB-image object module can be used) should be:

```
// JOB    CATALOG BUFFER IMAGE
// OPTION CATAL
LIBDEF *,SEARCH=IJSYSRS.SYSLIB,CATALOG=IJSYSRS.SYSLIB
  PHASE $$BUCBxx,*
  INCLUDE IJBxxxxx
   ...
   ...       PHASE and INCLUDE statements for additional
   ...       buffer-image phases.
   ...
// EXEC LNKEDT,PARM='MSHP'
/*
/&
```

On completion of this job step, you can load any of the newly cataloged buffer-image phases by entering an LFCB or LUCB command in the format:

```
LFCB cuu,phasename
LUCB cuu,phasename,NOCHK
```

Chapter 18. Extending and Tailoring System Files

Extending the VSE/ICCF DTSFILE

If necessary, you can allocate more space to the VSE/ICCF DTSFILE. You can do this by defining a larger extent on SYSWK1 or by defining extents on several volumes.

Estimating Used Space

Before you extend the amount of space reserved for the DTSFILE, you should make sure that you actually need to do so. Space allocated to the DTSFILE cannot be used for any other purpose.

"z/VSE Disk Layouts" in the manual *z/VSE Planning* documents the initial space allocations for the DTSFILE (ICCF.LIBRARY) on SYSWK1. You can estimate how much of that space is currently in use in two ways:

1. During system startup, look for the following message:
   ```
   K088I HI FILE RECORDS=number (nn%)
   ```
   This message is issued for the F2 (CICS-VSE/ICCF) partition. If **nn%** is near 0%, it may mean that the reserved space is almost used up.
2. Access the **DTSUTIL** utility of VSE/ICCF. From the *z/VSE Function Selection* panel select the *Command Mode*. Enter first *$DTSUTIL* to invoke the utility and then
   ```
   DISPLAY LIBRARY
   ```
   to display library information. The resulting display shows the total amount of DTSFILE space and the amount of free space left. The following example shows that at least 372,333 records of the space reserved for the libraries are still available:
   ```
   RECORDS IN FILE     432,040
   HI FILE RECORDS     372,333
   ```

It is also recommended that you back up and restore the VSE/ICCF libraries before you allocate more space to the DTSFILE. This may free up some space that currently cannot be used. *z/VSE Operation* has information about backing up and restoring the VSE/ICCF libraries under "Backing Up VSE/ICCF Libraries" and under "Restoring VSE/ICCF Libraries".

Using Skeleton SKDTSEXT

With the skeleton SKDTSEXT, you create a job that extends the VSE/ICCF DTSFILE to a multivolume, multiextent file. The skeleton is shipped in VSE/ICCF library 59. Figure 83 on page 317 shows the skeletons and the variables you must specify.

To extend the DTSFILE, you must perform the following steps:

1. Backup the DTSFILE on tape. Use the *Backup the ICCF Library on Tape* dialog.
2. Create a restore job for the DTSFILE using the *Restore the ICCF Library from Tape* dialog. Submit the job. Defer its execution by using disposition L.

Note: The restore job **must** be in the VSE/POWER reader queue, before you run the extend job. In this way, it can be released while VSE/ICCF is down. If the job is not there, you might later have to reinstall your system in order to use VSE/ICCF again.

3. Prepare the extend job by using skeleton SKDTSEXT. Refer to Figure 83 on page 317 for details of the skeleton and what needs to be observed when using it. Comments included in the skeleton are not shown.

   Submit the extend job with disposition L. As a result, the job is stored in the VSE/POWER reader queue but not released for processing.
4. Update the **label information** in the STDLABEL procedure. The reason is that the job you created (via SKDTSEXT) can only update the temporary label area on disk but not the STDLABEL procedure itself. To update the label information permanently, perform the following steps:
   a. Copy the STDLABEL procedure from IJSYSRS.SYSLIB to your primary VSE/ICCF library. For copying, use the VSE/ICCF LIBRP command. In command mode, enter:

      ```
      LIBRP IJSYSRS.SYSLIB STDLABEL.PROC STDLABEL
      ```
   b. Update the DTSFILE label information.
   c. Ensure that all necessary JCL statements are present and submit procedure STDLABEL for processing (cataloging).
5. Shutdown the system and perform a MINI startup. Release the jobs in the reader queue. Release first the extend job and after its successful completion the restore job.
6. To activate the changed characteristics of the DTSFILE, shutdown the system and perform a normal startup.

```
* $$ JOB JNM=ICCFEXT,CLASS=0,DISP=D
* $$ LST CLASS=A,DISP=D
      .
      .
// JOB ICCFFORM  FORMAT YOUR  z/VSE ICCF DTSFILE ON NEW EXTENTS
// DLBL DTSFILE,'ICCF.LIBRARY',99/366,DA
// EXTENT SYS010,SYSXXX,1,0,NNNNN,MMMMM
// EXTENT SYS011,SYSYYY,1,1,NNNNN,MMMMM
// ASSGN SYS010,DISK,VOL=SYSXXX,SHR
// ASSGN SYS011,DISK,VOL=SYSYYY,SHR
// PAUSE     BE SURE ICCF IS NOT OPERATIONAL
// EXEC DTSUTIL
FORMAT LIB(199) USERS(199)
/*
/&
// JOB UPDATE    UPDATE STDLABEL AREA  AND DTRICCF.PROC
// OPTION STDLABEL=DELETE
DTSFILE
/*
// OPTION STDLABEL=ADD
// DLBL DTSFILE,'ICCF.LIBRARY',99/366,DA
// EXTENT SYS010,SYSXXX,1,0,NNNNN,MMMMM
// EXTENT SYS011,SYSYYY,1,1,NNNNN,MMMMM
/*
      .
      .
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
CATALOG DTRICCF.PROC DATA=YES REPL=YES
// ASSGN SYS010,DISK,VOL=SYSXXX,SHR
// ASSGN SYS011,DISK,VOL=SYSYYY,SHR
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY DTRICCF.PROC REPLACE=YES
/*
/&
* $$ EOJ
```

Figure 83. Extending the VSE/ICCF DTSFILE (Skeleton SKDTSEXT)

Observe the following when using skeleton SKDTSEXT:

- First copy skeleton SKDTSEXT to your primary VSE/ICCF library.
- Assign a logical unit to every extent that you define. SYS010 must be the logical unit assigned to the first extent, SYS011 to the second extent, and so on. However, if you define the second extent on the same volume as the first extent, you must use SYS010 also for the second extent. Skeleton SKDTSEXT uses two extents on different volumes.

  Ensure that no overlap occurs with the extents of other files.
- SYS**XXX** and SYS**YYY** define the disk volume(s). SYSWK1, for example.
- **NNNNN** defines the beginning of an extent.
- **MMMMM** defines the total amount of space to be reserved for the extent.
- The disk volume used for the extent must have the SHR (share) option.
- In the skeleton, you also have to complete statements that update the system's standard labels and the procedure DTRICCF. DTRICCF contains the assignments for the DTSFILE. It is processed during startup of VSE/POWER and CICS.

After making the changes, run the DTRSEXIT macro. This macro deletes specific comments from the skeleton. You should do this before you file the skeleton. On the command line, enter:

```
@DTRSEXIT
```

Reformatting the VSE/ICCF DTSFILE

The DTSFILE generated by z/VSE defines 199 libraries and 199 VSE/ICCF user ID records. You can *reformat* the DTSFILE to create additional libraries and user ID records.

Notes:

1. Before changing the original definitions for libraries and user IDs, refer to "Formatting the Library File or Changing its Size" in the manual *VSE/ICCF Administration and Operation*. This manual has detailed information about formatting the DTSFILE.
2. For z/VSE, you *must* define VSE/ICCF libraries with the DATE option.

Skeleton SKICCFMT has the values that z/VSE specifies for the file. If you use the skeleton, first copy it from VSE/ICCF library 59 to your primary VSE/ICCF library. Then edit the copied file.

To reformat the DTSFILE, perform the following steps:

- Backup the DTSFILE on tape. Use the *Backup the ICCF Library on Tape* dialog.
- Create a restore job for the DTSFILE using the *Restore the ICCF Library from Tape* dialog. Replace line ″RESTORE ALL″ by ″RESTORE LIBRARIES(n) USERS(u) ALL″, and insert before the changed restore command a format command as follows:

  ```
  FORMAT LIBRARIES(n) USERS(u),
  ```

  where ″n″ is the number of desired libraries and ″u″ the number of desired VSE/ICCF users.
- Prepare the job to add new libraries according to skeleton SKICFFMT. Be aware that no format command is needed and only those libraries must be mentioned, that have not already been added when the DTSFILE backup was done. Insert a PAUSE statement before calling DTSUTIL to be able to disconnect the DTSFILE when running the job. Submit the job.

Figure 84 shows the skeleton. Comments included in the skeleton are not shown.

The sample has only one variable, **-V001-**. You can also change, add, or delete other statements or parameters.

```
* $$ JOB JNM=SKICFFMT,DISP=D,CLASS=0
// JOB SKICFFMT
// ASSGN SYS010,DISK,VOL=-V001-,SHR
// ASSGN SYS011,DISK,VOL=-V002-,SHR
// EXEC DTSUTIL
FORMAT LIBRARIES(199) USERS(199)
*  ADD LIBRARY 1  . . .
ADD LIBRARY FREESPACE(40) DATE

*  ADD LIBRARY 2  . . .
ADD LIBRARY FREESPACE(10) DATE

*  ADD LIBRARIES 3,4,5, AND 6  . . .
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE NOCOMMON PUBLIC
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE NOCOMMON PUBLIC
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE NOCOMMON PUBLIC
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE NOCOMMON PUBLIC
*  ADD LIBRARIES 7 THRU 49  . . .

ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE

               .
               .        (Additional ADD LIBRARY statements)
               .

ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
```

Figure 84. Skeleton SKICFFMT, Part 1 of 2 (Formatting the VSE/ICCF DTSFILE)

In the ASSGN statement, replace the variable **-V001-**. Specify the volume number of the disk where the DTSFILE resides.

The skeleton adds the following libraries:

1 = For VSE/ICCF administrator
2 = Common library
3-6 = Public libraries

Libraries 7 - 49 are private libraries that can be assigned to users.

Note: Libraries 8, 9, and 10 are used as primary libraries by the predefined z/VSE users OPER, PROG, and SYSA.

```
*  ADD LIBRARIES 50 THRU 68  . . .

ADD LIBRARY DATE NOCOMMON PUBLIC
ADD LIBRARY DATE NOCOMMON PUBLIC

            .
            .           (Additional ADD LIBRARY statements)
            .

ADD LIBRARY DATE NOCOMMON PUBLIC
ADD LIBRARY DATE NOCOMMON PUBLIC
*  ADD LIBRARIES 69 THROUGH 99

ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE

            .
            .           (Additional ADD LIBRARY statements)
            .

ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
*  ADD LIBRARIES 100 THROUGH 199.

ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE

            .
            .           (Additional ADD LIBRARY statements)
            .
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
DSERV ALL COMMON SORTED
END
/*
/&
* $$ EOJ
```

Figure 85. Skeleton SKICFFMT, Part 2 of 2 (Formatting the VSE/ICCF DTSFILE)

The skeleton adds libraries 50 - 68 which are reserved for z/VSE and are used by the Interactive Interface. The skeleton also adds libraries 69 - 99 which are private and can be assigned to users.

After making the changes, run the DTRSEXIT macro. This macro deletes specific comments from the skeleton. You should do this before you file the skeleton. On the command line, enter:

@DTRSEXIT

After the macro finishes, file the job. You can then submit it to the system for processing.

Extending VSE/POWER Files

If there is a need to extend the VSE/POWER files, do it carefully. Incorrect specifications are likely to cause startup problems. Two sample job streams (skeletons) are provided, the first one for extending both the data file (IJDFILE) and the queue file (IJQFILE), the second one for extending the data file (IJDFILE) only.

Extending the Queue File and Data File by a VSE/POWER Cold Start

The VSE/POWER IJQFILE is designed for about 500 to 1000 queue file records. Extending it or placing the queue and/or the data file at a different disk location can only be done by a VSE/POWER cold start.

Note: If it is necessary to increase the queue file, you must also increase the size of the partition GETVIS area (skeleton SKALLOCx). For the values to be specified, refer to “Planning for VSE/POWER” in the manual *VSE/POWER Administration and Operation*.

The following steps are required to extend/move the space for the VSE/POWER data file and queue file:

1. Use the POFFLOAD BACKUP command to save the queue entries on tape.
2. Update the file extent information in the label procedure STDLABEL.PROC. Proceed as follows:
   a. Copy the label procedure from IJSYSRS.SYSLIB to your primary VSE/ICCF library. For copying, use the VSE/ICCF command LIBRP. In Command Mode, enter:
   ```
   LIBRP IJSYSRS.SYSLIB STDLABEL.PROC STDLABEL
   ```
   b. Update the file extent information NNNNN (beginning of extent) and MMMMM (amount of space). Sample statements are shown below:
   ```
   // DLBL IJQFILE,'VSE.POWER.QUEUE.FILE',99/366,DA
   // EXTENT SYS001,DOSRES,1,0,NNNNN,MMMMM
   // DLBL IJDFILE,'VSE.POWER.DATA.FILE',99/366,DA
   // EXTENT SYS002,SYSWK1,1,0,NNNNN,MMMMM
   ```
   For a description of the DLBL and EXTENT statements, see the manual *z/VSE System Control Statements* under “DLBL” and under “EXTENT”.

   To extend the queue file or the data file, ensure that enough space is available on the disk volume(s) that you use. The queue file can have only one extent. The data file can have up to 32 extents, and all of its extents must reside on disk volumes of the same device type.

   Notes:
   1) If you extend the VSE/POWER data file (IJDFILE) over more than one volume, you have to use consecutive SYSnnn numbers starting with SYS002 and update the label information in STDLABEL.PROC accordingly.
   2) If you define extents on multiple volumes or if you move to a different volume than SYSWK1, you must update procedure DTRPOWR in IJSYSRS.SYSLIB. DTRPOWR includes the ASSGN statements for the VSE/POWER account, queue, and data files. Refer also to the skeleton SKPWREXT shown in Figure 86 on page 322.

3) If you move the VSE/POWER queue file (IJQFILE) to a different volume than DOSRES, you must update procedure DTRPOWR in IJSYSRS.SYSLIB.

c. Add to your edited member the control statements listed below. In front of your member add:

```
* $$ JOB JNM=RECAT,CLASS=0,PRI=9
// JOB RECAT
// LIBDEF *,CATALOG=IJSYSRS.SYSLIB
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
```

At the end of your member add:

```
/&
* $$ EOJ
```

3. Submit the updated label procedure by selecting option 7 or by entering: SUBMIT STDLABEL. This causes the procedure to be written back into IJSYSRS.SYSLIB, replacing the original label-information statements.
4. Shut down your system; follow the procedure described in the *VSE/ESA Operation* manual under "Shutting Down the System".
5. Perform a COLD startup of your sytem.
6. Reload the queue entries saved on tape. Use the POFFLOAD LOAD command. The manual *z/VSE Operation* describes in detail how to use the POFFLOAD command under “Offloading and Loading VSE/POWER Queues”. For a detailed description of the LIBRP command, refer to the manual *VSE/ICCF User's Guide* under “LIBRP Macro”.

You may replace steps 2 and 3 by using skeleton SKPWREXT provided in VSE/ICCF library 59 and follow its instructions.

```
* $$ JOB JNM=POWEREXT,CLASS=0,DISP=D
* $$ LST CLASS=A,DISP=D
// JOB POWEREXT
* -----------------------------------------------------------------------
*  STEP 1
* -----------------------------------------------------------------------
*  CHANGE THE LABEL PROCEDURE
*      STDLABEL.PROC IN IJSYSRS.SYSLIB AS FOLLOWS:
*        1. COPY THE PROCEDURE INTO YOUR PRIMARY LIBRARY USING LIBRP.
*        2. MODIFY THE LABELS FOR POWER DATA AND/OR  ACCOUNT
*           FILE AND SAVE THE MODIFIED FILE.
*           IF CHANGING THE QUEUE FILE IT MIGHT BE NECESSARY TO
*           ADJUST THE PARTITION SIZE, REFER ALSO TO THE
*           ADMINISTRATION GUIDE.
*        3. INSERT THE NAME OF THIS ICCF MEMBER IN THE SUBSEQUENT
*           INCLUDE STATEMENT - VARIABLE --V001--
*       OR USE DITTO AND CHANGE THE PROCEDURE DIRECTLY, DON'T
*       FORGET TO CHANGE ALSO IN PRD2.SAVE AND REMOVE FOLLOWING
*       STEP.
// EXEC LIBR,PARM='MSHP'
AC S=IJSYSRS.SYSLIB
/INCLUDE --V001--
// EXEC LIBR,PARM='MSHP'
CON S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY STDLABEL.PROC R=Y
/*
```

Figure 86. Skeleton SKPWREXT (Part 1 of 2)

```
* --------------------------------------------------------------------
*  STEP 2
* --------------------------------------------------------------------
*  CHANGE PROCEDURE DTRPOWR
*      IF ANY OF THE POWER FILES WAS MOVED TO A DIFFERENT VOLUME
*      THE ASSIGNMENT MUST ALSO BE CHANGED.
*      CHANGE THE PROCEDURE AS YOU CHANGED THE LABEL PROCEDURE,
*      THE NAME OF THE PROCEDURE HAS TO BE CHANGED IN THE SUBSEQUENT
*      INCLUDE STATEMENT - VARIABLE --V002--
*      OR USE DITTO AND CHANGE THE PROCEDURE DIRECTLY, DON'T
*      FORGET TO CHANGE ALSO IN PRD2.SAVE AND REMOVE FOLLOWING
*      STEP.
// EXEC LIBR,PARM='MSHP'
AC S=IJSYSRS.SYSLIB
/INCLUDE --V002--
// EXEC LIBR,PARM='MSHP'
CON S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY DTRPOWR.PROC R=Y
/*
* --------------------------------------------------------------------
*  STEP 3
* --------------------------------------------------------------------
*  MAKE SURE A POWER COLD START IS PERFORMED
* -------------------------------------------------------------
// SETPARM XNCPU=' '
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;'
   SET XPWMODE=COLD
/*
* -------------------------------------------------------------
*  STEP 4
* -------------------------------------------------------------
*  FOR THE FOLLOWING COLD START, BACKUP THE DATA FILE AS FOLLOWS:
*             1. REPLY "(END/ENTER)" TO FINISH THIS JOB
*             2. SHUTDOWN ALL PARTITIONS EXCEPT POWER
*             3. POFFLOAD YOUR POWER QUEUES
*                  POFFLOAD BACKUP,ALL,CUU
*                                      CUU OF THE TAPE DRIVE
*             4. IPL FROM DOSRES, SYSTEM WILL ISSUE A COLD START
*             5. WHEN VSE/POWER IS UP AFTER IPLING, LOAD THE DATA
*                BACK INTO THE SYSTEM:
*                  POFFLOAD LOAD,ALL,CUU
*                              CUU OF THE TAPE DRIVE
* -------------------------------------------------------------
// PAUSE
/&
* $$ EOJ
```

Figure 86. Skeleton SKPWREXT (Part 2 of 2)

After making the changes, run the DTRSEXIT macro. This macro deletes specific comments from the skeleton. You should do this before you file the skeleton. On the command line, enter:

@DTRSEXIT

After the macro finishes, file the job. You can then submit it to the system for processing.

Extending the Data File during a VSE/POWER Warm Start

If there is only a need for more VSE/POWER data file space, for example to be able to store more LST output, you may increase the data file during a VSE/POWER Warm start. This is possible without reformatting the existing queue file and data file extents as required in case of a Cold start. Therefore, the Data File extension will **not** affect the already spooled data and cause no long system-down-time.

To trigger Data File extension during Warm start, append **one** extent with ascending sequence number to the existing IJDFILE DLBL/EXTENT statements. The new extent must be added as the last extent, because VSE/POWER accesses the extents as a contigious stream of DBLKs, starting with DBLK #0 and ending with DBLK #n. DBLKs on existing extents are already referred to by their number which cannot be changed. VSE/POWER will detect the appended extent during the next Warm start and will ask the operator to confirm Data File extension by:

```
1QD2D DATA FILE EXTENT NO. mm FOUND - TO FORMAT REPLY 'YES' ELSE 'NO'
(// EXTENT SYSxxx,volid,1,nnn,start,length)
```

When the operator replies YES, VSE/POWER verifies the specified location of the appended extent. If the new extent is accepted, formatting of the new extent takes place after Warm start has been completed. While the additional extent is formatted, spooling is already enabled. More details about this process can be found in the manual *VSE/POWER Administration and Operation*.

Note: As long as the maximum number of Data File extents is not yet reached, "Data File extension during Warm start" can be repeated during a subsequent VSE/POWER Warm start. Therefore, it is recommended to define a Queue File larger than needed during a VSE/POWER Cold start to avoid any further VSE/POWER Queue File extensions. The following steps are required:

1. Update the file extent information in the label procedure STDLABEL.PROC as follows:
   a. Copy the label procedure from IJSYSRS.SYSLIB to your primary VSE/ICCF library. For copying, use the VSE/ICCF command LIBRP. In *Command Mode*, enter:
   ```
   LIBRP IJSYSRS.SYSLIB STDLABEL.PROC STDLABEL
   ```
   b. Append another EXTENT statement for the data file (IJDFILE). Sample statements show DLBL, EXTENT, and ASSGN statements before and after appending a new EXTENT. The old DLBL/EXTENT and ASSGN statements have the following values:
   ```
   // DLBL IJDFILE,'VSE.POWER.DATA.FILE',99/366,DA
   // EXTENT SYS002,SYSWK1,1,0,start1,length1
   // EXTENT SYS002,SYSWK1,1,1,start2,length2
   // EXTENT SYS003,SYSWK4,1,2,start3,length3

   // ASSGN SYS002,DISK,VOL=SYSWK1,SHR      POWER DATA FILE 1 + 2
   // ASSGN SYS003,DISK,VOL=SYSWK4,SHR      POWER DATA FILE 3
   ```
   The new DLBL/EXTENT and ASSGN statements have the following values:
   ```
   // DLBL IJDFILE,'VSE.POWER.DATA.FILE',99/366,DA
   // EXTENT SYS002,SYSWK1,1,0,start1,length1
   // EXTENT SYS002,SYSWK1,1,1,start2,length2
   // EXTENT SYS003,SYSWK4,1,2,start3,length3
   // EXTENT SYS004,SYSWK2,1,3,start4,length4
   ```

```
// ASSGN SYS002,DISK,VOL=SYSWK1,SHR      POWER DATA FILE 1 + 2
// ASSGN SYS003,DISK,VOL=SYSWK4,SHR      POWER DATA FILE 3
// ASSGN SYS004,DISK,VOL=SYSWK2,SHR      POWER DATA FILE 4
```

For a description of the DLBL and EXTENT statements, see the manual *z/VSE System Control Statements* under "DLBL" and under "EXTENT".

To extend the Data File, ensure that enough space is available on the disk volume(s) that you use. The Data File can have up to 32 extents, and all these extents must reside on disk volumes of the same device type.

Notes:

1) If you extend the VSE/POWER data file (IJDFILE) over more than one volume, you have to use consecutive SYSnnn numbers starting with SYS002 and update the label information in STDLABEL.PROC accordingly.
2) If you define extents on multiple volumes or if you move to a different volume than SYSWK1, you must update procedure DTRPOWR in IJSYSRS.SYSLIB. DTRPOWR includes the ASSGN statements for the VSE/POWER account, queue, and data files. Refer also to skeleton SKPWRDAT shown in Figure 87 on page 326.

c. Add to your edited member the control statements listed below.

   In front of your member:

```
* $$ JOB JNM=RECAT,CLASS=0,PRI=9
// JOB RECAT
// LIBDEF *,CATALOG=IJSYSRS.SYSLIB
// EXEC LIBR,PARM='MSHP'
ACC S=IJSYSRS.SYSLIB
```

   At the end of your member:

```
/&
* $$ EOJ
```

2. Submit the updated label procedure by selecting option 7 or by entering:

```
SUBMIT STDLABEL
```

   This causes the procedure to be written back into IJSYSRS.SYSLIB, replacing the original label-information statements.
3. Shut down your system; follow the procedure described in the *z/VSE Operation* manual under "Shutting Down the System".
4. Re-IPL your system which will prompt you with message 1QD2D.

For a detailed description of the LIBRP command, refer to the manual *VSE/ICCF User's Guide* under "LIBRP Macro".

You may replace steps 1 and 2 by using skeleton SKPWRDAT provided in VSE/ICCF library 59 and follow its instructions.

```
* $$ JOB JNM=POWERDAT,CLASS=0,DISP=D
* $$ LST CLASS=A,DISP=D
// JOB POWERDAT
* ----------------------------------------------------------------------
*  STEP 1
* ----------------------------------------------------------------------
*  CHANGE THE LABEL PROCEDURE
*      STDLABEL.PROC IN IJSYSRS.SYSLIB AS FOLLOWS:
*        1. COPY THE PROCEDURE INTO YOUR PRIMARY LIBRARY USING LIBRP.
*        2. APPEND ONE EXTENT FOR POWER DATA FILE IJDFILE AND
*           SAVE THE MODIFIED FILE.
*     NOTE: THE ADDITIONAL EXTENT MUST EITHER RESIDE ON THE SAME
*           DISK AS THE LAST EXTENT AND USE THE SAME LOGICAL UNIT
*           NUMBER (SYSNNN) OR MUST RESIDE ON A DISK
*           CONTAINING NO DATA FILE EXTENTS SO FAR AND THE LOGICAL
*           UNIT NUMBER SYSNNN MUST BE INCREMENTED BY ONE.
*
*        3. INSERT THE NAME OF THIS ICCF MEMBER IN THE SUBSEQUENT
*           INCLUDE STATEMENT - VARIABLE --V001--
*       OR USE DITTO AND CHANGE THE PROCEDURE DIRECTLY, DON'T
*       FORGET TO CHANGE ALSO IN PRD2.SAVE AND REMOVE FOLLOWING
*       STEP.
// EXEC LIBR,PARM='MSHP'
AC S=IJSYSRS.SYSLIB
/INCLUDE --V001--
// EXEC LIBR,PARM='MSHP'
CON S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY STDLABEL.PROC R=Y
/*
* ----------------------------------------------------------------------
*  STEP 2
* ----------------------------------------------------------------------
*  CHANGE PROCEDURE DTRPOWR
*      IF THE APPENDED EXTENT OF THE POWER DATA FILE RESIDES ON
*      A NOT YET ASSIGNED VOLUME, YOU MUST ADD THE ASSIGNMENT.
*      CHANGE THE PROCEDURE AS YOU CHANGED THE LABEL PROCEDURE,
*      THE NAME OF THE PROCEDURE HAS TO BE CHANGED IN THE SUBSEQUENT
*      INCLUDE STATEMENT - VARIABLE --V002--
*      OR USE DITTO AND CHANGE THE PROCEDURE DIRECTLY, DON'T
*      FORGET TO CHANGE ALSO IN PRD2.SAVE AND REMOVE FOLLOWING
*      STEP.
// EXEC LIBR,PARM='MSHP'
AC S=IJSYSRS.SYSLIB
/INCLUDE --V002--
// EXEC LIBR,PARM='MSHP'
CON S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY DTRPOWR.PROC R=Y
/*
* ----------------------------------------------------------------
*  STEP 3
* ----------------------------------------------------------------
*  DURING THE FOLLOWING WARM START, VSE/POWER WILL EXTEND THE DATA FILE
*             1. REPLY "(END/ENTER)" TO FINISH THIS JOB
*             2. IPL FROM DOSRES, SYSTEM WILL ISSUE A WARM START
*             3. WHEN VSE/POWER REQUESTS CONFIRMATION FOR
*                DATA FILE EXTENSION BY MESSAGE 1QD2D, REPLY 'YES'
* ----------------------------------------------------------------
// PAUSE
/*
/&
* $$ EOJ
```

Figure 87. Skeleton SKPWRDAT

After making the changes, run the DTRSEXIT macro. This macro deletes specific comments from the skeleton. You should do this before you file the skeleton. On the command line, enter:

```
@DTRSEXIT
```

After the macro finishes, file the job. You can then submit it to the system for processing.

Chapter 19. Tailoring Terminal Functions and Console Definitions

This chapter describes several tasks for tailoring terminal functions. For the task of changing z/VSE console definitions, refer directly to “Tailoring Console Definitions” on page 337.

Using Skeleton IESxLOGO

The **IESxLOGO** skeleton allows you to modify the *z/VSE Online* panel and related functions. With this panel you sign on to z/VSE. The ’**x**’ in the logo name refers to the language being used:

E English

G German

S Spanish

J Japanese

An example of this panel is shown in Figure 1 on page 5.

Using IESxLOGO (where ’x’ refers to the language you are using: E, G, S, or J) you can:

1. Change the **logo** that is displayed on the panel. The default logo is z/VSE. You can implement your own logo design for the panel display.
2. Set a **limit** for invalid **sign-on** attempts.
3. Allow **every** CICS user to **escape** to CICS from the panel without signing on to the Interactive Interface.

   To implement the escape function, you can either:
   - Specify that PF6 and PF9 are displayed on the panel. These PF keys are used for the escape facility.
   - Specify a 1 - 4 character string for the escape facility. A user can then enter this character string from the *z/VSE Online* panel to escape to CICS.

   Note: The security functions of the Interactive Interface (user ID, password) are bypassed when allowing to “escape” to CICS and use it in native mode.
4. Specify the offset or **cuu** for non-SNA terminals in the **netname** to use the PF3 function key when running under VM.
5. Configure the ″logon here″ function.

IESxLOGO modifications become effective for all the terminals defined to a CICS subsystem.

The skeleton is shipped in VSE/ICCF library 59. First copy it to your VSE/ICCF primary library and edit the copied skeleton. Refer to “Copying Skeletons” on page 6 for information on copying skeletons.

Figure 88 on page 330 through Figure 90 shows the skeleton. A description of the statements and changes follows each part of the skeleton.


```
* $$ JOB JNM=IESELOGO,CLASS=A,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB IESELOGO ASSEMBLE
// LIBDEF *,CATALOG=PRD2.CONFIG
* IN CASE GENERATION FEATURE IS INSTALLED ACTIVATE THE FIRST LIBDEF
* // LIBDEF SOURCE,SEARCH=(PRD2.GEN1,PRD1.BASE,PRD1.MACLIB)
// LIBDEF SOURCE,SEARCH=(PRD1.BASE,PRD1.MACLIB)
// OPTION CATAL,LIST
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAX*
               -200K,ABOVE)'
           .
           .
         GBLB  &PUNCH                SHALL WE PUNCH A CATALOG STATEMENT?
&PUNCH   SETB  1                     THIS TIME THE ANSWER IS         YES
         AIF   (NOT &PUNCH).BYPUN    IF NO CATALOG STATEMENT REQUIRED
         PUNCH ' CATALOG IESELOGO.OBJ REP=YES'
         PUNCH ' PHASE  IESELOGO,S'
.BYPUN   ANOP                        NO CATALOG STATEMENT REQUIRED
LOGO     TITLE 'z/VSE --  USER CHANGEABLE LOGO PHASE'
IESELOGO CSECT
         DC    CL8'IESLOGO'        MODULE IDENTIFIER
         DC    X'64'               VSE/ESA 2.4.0 AND HIGHER
         DC    AL1(LOGOLINS)       NUMBER OF LINES OF LOGO TEXT
         DC    H'0'                ...  RESERVED  ...
         DC    A(LOGOBA)           ADDRESS OF THE LOGO TEXT
         DC    A(ESCAPESW)         ADDRESS OF THE ESCAPE SWITCH
         DC    A(MAXNUMSO)         ADDRESS MAX. NUMBER SIGNON ATTEMPTS
         DC    A(0)                *
         DC    A(UCESCSTR)         ADDRESS OF THE UPPER CASE ESCAPE
*                                     CHARACTER STRING
         DC    A(MCESCSTR)         ADDRESS OF THE MIXED CASE ESCAPE
*                                     CHARACTER STRING
         DC    A(CUUOFFS)          ADDRESS OF CUU OFFSET IN NETNAME
         DC    A(SIGNONH)          ADDRESS OF SIGNON-HERE SWITCH
*----------------------------------------------------------------------*
*          THE LINES ABOVE THIS BOX MUST NOT BE CHANGED               *
*----------------------------------------------------------------------*
         SPACE 2
           .
         SPACE 2
           .
         EQU   70                            FIXED LENGTH OF EACH LINE
           .
         SPACE 2
           .
         SPACE 2
```

Figure 88. IESELOGO Skeleton, Part 1 of 3

Notes:

1. The statement

   ```
   // EXEC ASMA90....
   ```

   calls the High Level Assembler. Refer to the manual *z/VSE Guide to System Functions* under "High Level Assembler Considerations" for further details.
2. Do **not** change the statements in this part of the skeleton, except for the case described below:
   You may have a system with multiple CICS partitions and for each of those you want individual logos displayed. The logo created with this skeleton is cataloged in library PRD2.CONFIG. If you want to create a second logo, you must change the library definition (PRD2.CONFIG) in the LIBDEF statement. Otherwise, the second logo simply replaces the logo you created first. Choose

an appropriate sublibrary of your installation. Change the library search chain of the related CICS so that the sublibrary with the logo is early or first in the search chain.

Changing the LOGO Design

You use this section of the skeleton to change the logo design.

```
LOGOBA   EQU   *                 THIS LABEL MUST PRECEDE YOUR LOGO TEXT
*---------------------------------------------------------------------*
LOGOSKEL EQU   *                 THE SKELETON LOGO BEGINS HERE
*DC CL(L)'?????????????????????????????????????????????????????????????*
               ?????????'
*DC CL(L)'??????????????? YOU CAN REPLACE THE AREA ????????????????????*
               ?????????'
*DC CL(L)'?????????????? FILLED WITH QUESTION MARKS ???????????????????*
               ?????????'
*DC CL(L)'??????????????? WITH YOUR OWN LOGO TEXT ?????????????????????*
               ?????????'
*DC CL(L)'?????????????????????????????????????????????????????????????*
               ?????????'
*DC CL(L)'?????????????????????????????????????????????????????????????*
               ?????????'
*DC CL(L)'?????????????????????????????????????????????????????????????*
               ?????????'
*DC CL(L)'?????????????????????????????????????????????????????????????*
               ?????????'
*DC CL(L)'?????????????????????????????????????????????????????????????*
               ?????????'
*DC CL(L)'?????????????????????????????????????????????????????????????*
               ?????????'
*---------------------------------------------------------------------*
LOGOEND  EQU   *                  THIS LABEL MUST FOLLOW YOUR LOGO TEXT
```

Figure 89. IESELOGO Skeleton, Part 2 of 3

The LOGOBA label **must** be before the logo text and should **not** be changed. In the *DC CL(L) statements:

- Replace the question marks (**?**) with your own logo design.
- Replace the asterisks (*****) in column 1 with blanks for each DC statement that you use.

You can replace the question marks with text, block letters, or blank lines. Do **not** change the format of the skeleton; that is, the beginning and ending columns of the lines. The format follows the rules of Assembler language coding. If you change the format, there may be assembly errors or the sign-on program itself may not work correctly.

The LOGOEND label **must** follow the logo text. To implement the change, proceed as follows:

1. Submit skeleton IESxLOGO (where 'x' refers to the language you are using: E, G, S, or J) for processing.
2. Restart CICS.

Setting a Limit for Invalid Sign-On Attempts

A user without a valid password can try to gain access to the system again and again. To limit the chances of gaining unauthorized access through this trial-and-error method, you can restrict the number of unsuccessful sign-on attempts. If the limit is reached, z/VSE revokes the user ID to prevent further use. System administrator authority is required to use the dialog for resetting a revoked user ID. Refer to "Resetting a Revoked User ID" on page 127 for further details.

Note: You can also use the IESIRCVT program to set a limit for invalid sign-on attempts. For details, see "Changing the Minimum Password Length and Revoke Details" on page 127.

To implement a limit for sign-on attempts, you must locate line MAXNUMSO in skeleton IESELOGO shown in Figure 90. In this line change the value of 5 (H'5'), which is the default value, into the number of attempts you want to allow, for example 3 (H'3'). You can specify any number from 1 through 9999. 0 means that no limit is set.

```
LOGOLINS EQU   (LOGOEND-LOGOBA)/L  NUMBER OF LOGO TEXT LINES
         SPACE 3
           .
MAXNUMSO DC    H'5'               MAX. NUMBER INVALID SIGNON ATTEMPTS
         SPACE 1
           .
SIGNONH DC     C'Y'               SIGNON-HERE CAPABILITY
         SPACE 1
           .
         SPACE 1
           .
ESCAPESW DC    C'N'               ESCAPE SWITCH
         SPACE 2
           .
UCESCSTR DC    CL4'    '           THIS IS THE CHARACTER STRING THE
*              TERMINAL OPERATOR SHOULD KEY INTO  . . .
*
*
           .
MCESCSTR DC    CL4'    '           THIS IS THE CHARACTER STRING THE
*              TERMINAL OPERATOR SHOULD KEY INTO  . . .
           .
         SPACE 2
           .
CUUOFFS  DC    H'1'               CUU OFFSET (0-5) IN NETNAME
         SPACE 2
*----------------------------------------------------------------------*
         END   ,                   NOTE --> NO LABEL ON END CARD
/*
// EXEC LNKEDT
/*
/&
* $$ EOJ
```

Figure 90. IESELOGO Skeleton, Part 3 of 3

To implement the change, proceed as follows:

1. Submit skeleton IESELOGO for processing.
2. Restart CICS.

Note: If you change the value of MAXNUMSO, you must then run program IESIRCVT to activate this change.

Controlling the Escape Facility

With line ESCAPESW in skeleton IESELOGO (shown in Figure 90 on page 332), you can control whether PF6 and PF9 are displayed on the sign-on panel. These two PF keys are used for the CICS escape facility. A user can press PF6 or PF9 to escape to native CICS without signing on to the Interactive Interface. In this case, security values for z/VSE and CICS are **not** established.

If you want to have the escape facility with the PF keys, change the N value in line ESCAPESW to **Y**.

If you have terminals which do not have PF6 and PF9, you can specify a 1 - 4 character string for the escape facility. You would do the following:

1. Change the N value in line ESCAPESW to **Y**.
2. In the following statements, insert a 1 - 4 character string between the single quotes (’ ’):
   - UCESCSTR DC CL4’ ’
   - MCESCSTR DC CL4’ ’

   UCESCSTR is for escape with uppercase (equivalent to PF6). MCESCSTR is for escape with mixed case (equivalent to PF9). Transaction IDs are translated in uppercase (UCTRANID).

   You can use special characters, but you cannot specify lowercase letters. If your character string is shorter than four characters, it must be padded with blanks on the right.

After filing the skeleton, perform the following steps to implement your changes:

1. Submit the completed skeleton for processing. The skeleton assembles and catalogs the logo module.
2. Check for any errors in the assembly. Correct any errors before proceeding.
3. Restart CICS.

Specifying cuu in Netname

According to the naming convention for non-SNA terminals, the generated VTAM netname contains the **cuu** (channel and unit number) in position 2-4:

x**cuu**xxxx

This is needed by z/VSE when running under VM to offer the PF3 function key (RETURN TO VM) on the *z/VSE Online* panel. If you use your own naming convention, you can specify the position of cuu using the IESxLOGO (where ’x’ refers to the language you are using: E, G, S, or J) skeleton.

Perform the following steps:

1. Locate line CUUOFFS in skeleton IESxLOGO (shown in Figure 90 on page 332). In this line, change the default offset 1 (H’1’) to the offset at which *cuu* starts in *netname*.
2. Submit skeleton IESxLOGO for processing.
3. Restart CICS.

Configure ’Logon Here’

If a user wants to sign-on to a z/VSE system but is already signed-on at another terminal, the message

```
USER ID 'xxxxx    ' IS ALREADY IN USE AT TERMINAL 'nnnn'
```

is displayed and the entry panel provides a PF key (PF12) with the function LOGON HERE.

LOGON HERE specifies that if this user ID is already logged on, it should be disconnected from its current terminal and reconnected at the terminal where this logon is requested.

If you want to disable this function, change the statement SIGNONH in skeleton IESxLOGO (where 'x' refers to the language you are using: E, G, S, or J) to **N** (No):

```
   ⋮
SIGNONH DC     C'N'              SIGNON-HERE CAPABILITY
   ⋮
```

```
IESADMSO1                        z/VSE ONLINE
    5609-ZVS and Other Materials (C) Copyright IBM Corp. 2004 and other dates


                                ++
                               ++    VV    VV    SSSSS    EEEEEEE
                zzzzzz        ++     VV    VV   SSSSSSS   EEEEEEE
                zzzzz        ++      VV    VV  SS         EE
                   zz       ++       VV    VV   SSSSSS    EEEEEE
                  zz       ++        VV    VV    SSSSSS   EEEEEE
                zzzzzz    ++          VV  VV         SS  EE
               zzzzzzz   ++            VVVV     SSSSSSS   EEEEEEE
                                        VV       SSSSS    EEEEEEE

        Your terminal is A001 and its name in the network is D3010001
        Today is 08/24/2004   To sign on to DBDCCICS  --  enter your:

       USER-ID........ ________   The name by which the system knows you.
       PASSWORD.......            Your personal access code.

PF1=HELP       2=TUTORIAL   3=TO VM      4=REMOTE APPLICATIONS     6=ESCAPE(U)
                            9=Escape(m) 10=NEW PASSWORD
```

Figure 91. Logon Here Panel

Recovering Terminal Connections

When a terminal is switched off without signing off or loses its VTAM connection to the CPU, the Interactive Interface is unaware of it. The user ID and the related control blocks are not freed. This prevents a signing on with the same user ID from another terminal.

To help avoid such situations, z/VSE provides the program IESCLEAN. This program frees all the resources related to specific user ID and performs a sign off from the Interactive Interface. A user can then sign on again with the freed user ID from a different terminal but without the reconnect facility available.

Implementing Program IESCLEAN

Programming Interface information

z/VSE uses the CICS node error program DFHZNEP to provide a link to program IESCLEAN. Refer to the manual *CICS Customization Guide* for details on node error processing.

In VSE/ICCF library 59, z/VSE provides three sample programs to modify program IESCLEAN:
IESZNEP
IESZNEPS
IESZNEPX

Program IESCLEAN is invoked if one of the following CICS error codes occur:
10 Node not activated
49 Node session terminated
57 Terminal released by master terminal operator
61 (with Sense=0831) POWER OFF at SNA terminals.
A7 Bracket error
D1 Node unrecoverable

You can add or remove error codes as needed by your installation.

You can use the sample programs provided as follows:

- **IESZNEP** includes a complete node error program (NEP) DFHZNEP which is active by default. By changing and submitting skeleton IESZNEP, you can replace NEP DFHZNEP.
- If you currently operate with a user supplied node error program, you can either use sample IESZNEPS or sample IESZNEPX:
  - Sample **IESZNEPS**:

    This sample assumes that you use the CICS sample node error program (DFHNEPS macro).

    IESZNEPS contains an error processor to be included into an existing sample NEP with the statement: COPY IESZNEPS. When being submitted, the sample creates member IESZNEPS.A and stores it in library PRD2.CONFIG. The group number assigned to the error processor must be unique.
  - Sample **IESZNEPX**:

    This sample assumes that you use your own user written node error program.

    IESZNEPX contains an error processor to be included into an existing user written NEP with the statement: COPY IESZNEPX. When being submitted, the sample creates member IESZNEPX.A and stores it in library PRD2.CONFIG.

To modify program IESCLEAN, the following steps are required:

1. Select one of the sample programs according to the needs of your installation.
2. Submit the sample program for processing.
3. Prepare your own node error program (if necessary).
4. Assemble your node error program (if necessary).
5. Restart CICS.

End of Programming Interface information

Signing On to Different CICS System

Programming Interface information

The sign-on exit **IESEXIT** (skeleton SKEXIT1) allows a user to sign on to different CICS subsystems in the same z/VSE system with the same user ID and password and receive different initial selection or application panels.

The initial selection panel or application name is defined for a given user in the user profile. In a z/VSE system with several CICS subsystems, the users are defined in the same control file meaning that for all CICS systems the same initial selection panel or application name is selected for a user at sign on. By means of this exit, you are able to specify an initial selection panel or application name depending on different CICS subsystems which use the same control file.

The program needs to be defined in the CICS CSD file (CEDA). Control must be given back via CICS RETURN.

The following parameters are provided through the CICS communication area (COMMAREA):

```
* PARAMETER LIST FOR SIGNON EXIT
PARMX    DS    0H                  PARAMETER LIST FOR EXIT1
APPLIDX  DS    CL8                 APPLICATION ID (padded with X'40's)
USERIDX  DS    CL8                 USERID (padded with X'00's)
CURRSELX DS    CL8                 CURRENT SELECTION/APPLICATION
NEWSELX  DS    CL8                 NEW SELECTION/APPLICATION
CURRTYPX DS    CL1                 CURRENT TYPE SELECTION/APPLICATION
NEWTYPX  DS    CL1                 NEW TYPE SELECTION/APPLICATION
PARMLENX EQU   *-PARMX             LENGTH OF PARAMETER AREA
```

Following is an example of how to retrieve a parameter in IESEXIT1:

```
         OC    EIBCALEN,EIBCALEN   COMMUNICATION AREA PROVIDED ?
         BZ    RETURN              NO, RETURN
         L     R1,DFHEICAP         GET POINTER TO COMMUNICATION AREA
         USING PARMX,R1
```

The parameters have the following meaning:

APPLIDX: ID of CICS as DBDCCICS, PRODCICS or any other.

USERIDX: User ID as defined for z/VSE.

CURRSELX: Name of initial selection panel or application as defined in the user profile.

NEWSELX: Name of initial selection panel or application to be used for this sign on. These names must be defined to the system.

CURRTYPX: Current type which can be selection (S) or application (A).

NEWTYPX: New type which can be selection (S) or application (A).

You should catalog program IESEXIT into sublibrary PRD2.CONFIG.

End of Programming Interface information

Tailoring Console Definitions

Programming Interface information

z/VSE uses predefined *console definitions* which you may modify if required. Note that these definitions are used for **all** active consoles at your z/VSE installation. Individual console tailoring is not possible.

Consult also the manual *z/VSE Planning* under "Console Support" before you start tailoring console definitions. The manual has a chapter which introduces and provides an overview of the console support provided.

Console definitions define:
- Panel data (fixed text displayed on the panel)
- PF key settings
- Local messages (related to the *Console* dialog)

The console definitions are shipped in source format as well as in phase format. The source of the console definitions is available as member IJBxDEF.Z, the corresponding phase is $IJBxDEF.PHASE. x defines one of the following languages:
E = English
G = German
J = Japanese
S = Spanish

The basic version of the system contains:
IJBEDEF.Z and **$IJBEDEF.PHASE** (English)

The following NLS (National Language Support) versions (source and phase) are also available:
IJBGDEF.Z and $IJBGDEF.PHASE (German)
IJBJDEF.Z and $IJBJDEF.PHASE (Japanese)
IJBSDEF.Z and $IJBSDEF.PHASE (Spanish)

$IJBEDEF.PHASE is always part of the system independent of the language ordered. Member IJBxDEF.Z contains definitions for:
- Panel data (either in English or in a national language).
- PF keys (either in English or in a national language).
- Local messages consisting of two entries each (the first entry specifies the local message text in English and the second one in a national language).

In order to modify console definitions, you must proceed as follows:
1. Edit the source in corresponding member IJBxDEF.Z.
2. Copy the member first from IJSYSRS.SYSLIB to your primary VSE/ICCF library. Use the VSE/ICCF command LIBRP.
3. Assemble the edited member and catalog the resulting object module as phase $IJBxDEF.PHASE into the system library IJSYSRS.SYSLIB.

The input required to create phase $IJBxDEF consists of multiple invocations of the macro IJBDEF as shown in Figure 92 on page 341.

Using Macro IJBDEF

The IJBDEF macro can be used to create entries for:

- Panel data
- PF key settings
- Local messages.

The macro does extensive validity and syntax checking for each macro definition. Every invocation of macro IJBDEF creates one entry.

Note

When reading this macro description refer to Figure 92 on page 341 for easier understanding. The figure includes coded examples of the different types of entries.

The IJBDEF macro has the following general format:

```
label    IJBDEF  parameters
```

The parameters are positional. Code a comma for any parameter that you omit (except for the last one).

The first parameter determines the type of definition:

PANEL Defines a panel data table entry

PFKEY Defines a PF key table entry

MSG Defines a local message text table entry

GEN Starts the table generation.

Note: The IJBDEF macro has no default for the first parameter.

Defining Panel Data

For defining panel data, the macro has the following format:

```
IJBDEF  PANEL,type,,,'edata','ndata'
```

One invocation defines an entry in the panel data table which consists of 30 entries allocated in IJBxDEF. The meaning of each parameter is as follows:

type Specifies the type of the particular panel data table entry. The constants to be specified for type are predefined and **cannot** be changed.

'edata' A string of data in English, enclosed in apostrophes, to be displayed at a defined position on the panel.

'ndata'

A string of data in a national language, enclosed in apostrophes, to be displayed at a defined position on the panel.

Note: In case 'ndata' is identical with 'edata', the specification of '=' is sufficient for 'ndata'.

The maximum data length that can be displayed varies with the panel type. The following list shows the maximum number of characters allowed for each panel type:

```
TPANL     10
TTITL     20
TSYST      8
TAPPL      8
TTIME      6
TPAGE      8
TTIMX      8
TACMD      3
TCMD     126
TDMSG     78
TFILT      8
TFILTR     8
TACTM      8
TFILX     17
THOLD      8
TNOHLD     8
TDIRC      6
TNUM1      8
TPAUSE     6
TNUM2      4
TIMSG      8
TMESG      8
TSUSP      8
TMODE      6
TMODXC    14
TMODXCM   14
TMODXCD   14
TMODXR    14
TMODXE    14
TMODXH    14
```

Defining PF Key Settings

For defining PF key settings, the macro has the following format:

```
IJBDEF  PFKEY,n,m,'etext','ntext','command'
```

The meaning of each parameter is as follows:

n PF key number in the range 1-12, or ENTER, or CLEAR

m One of the modes under which the PF key is valid:
- **C** console mode
- **R** redisplay mode
- **E** explain mode
- **H** help mode

'etext' The descriptive text in English to be displayed on the panel, with a maximum length of 8 characters, enclosed in apostrophes.

'ntext' The descriptive text in a national language to be displayed on the panel, with a maximum length of 8 characters, enclosed in apostrophes.

Notes:

1. The total length of the descriptive text for the PF keys (1-12) per mode ('etext' mode, 'ntext' mode) must not exceed 80 characters. For each single specification one extra space must be counted.
2. In case 'ntext' is identical with 'etext', the specification of '=' is sufficient for 'ntext'. See also Figure 92 on page 341.

'command'

A string of data being a local command, a z/VSE command or any other data such as a reply. It must be enclosed in apostrophes and may consist of up to 10 substrings separated from each other by 2 apostrophes.

Defining a Local Message Text

This function can be used when writing your own console applications. For defining the text of a local message, the macro has the following format:

```
Mn       IJBDEF  MSG,'etext','ntext'
```

The meaning of each parameter is as follows:

Mn
n is a number between 1 and 256, each of which specifies a particular message table entry. The number **cannot** be changed for system-supplied local messages. These are the numbers M1 through M80 which are reserved for use by z/VSE.

'etext'
Text of local message in English enclosed in apostrophes.

'ntext'
Text of local message in a national language enclosed in apostrophes.

Note: In case 'ntext' is identical with 'etext', the specification of **'='** is sufficient for 'ntext'. See also Figure 92 on page 341.

Starting Table Generation

For starting the generation of console definition tables, the macro has the following format:

```
         IJBDEF GEN,phase
```

IJBDEF GEN must be the last specification in a sequence of IJBDEF specifications. The variable "phase" specifies the name of the phase to be generated:

$IJBEDEF (English, which is the default)
$IJBGDEF (German)
$IJBJDEF (Japanese)
$IJBSDEF (Spanish)

Member IJBEDEF.Z

Figure 92 on page 341 shows member IJBEDEF.Z stored in system library IJSYSRS.SYSLIB. Note that the contents of IJBEDEF.Z as shown reflects the status at the time the manual was printed. The actual contents shipped may differ from the one shown in Figure 92.

The following job stream example can be used to reassemble member IJBEDEF.Z if tailoring is required:

```
* $$ JOB JNM=IJBEDEF,DISP=D,CLASS=Z,LDEST=(*,user)
* $$ LST CLASS=X,DISP=D
// JOB CATAL IJBEDEF CONSOLE DEFINITIONS
// OPTION NOSYSDUMP
// ID USER=FORSEC,PWD=FORSEC
// LIBDEF *,CATALOG=IJSYSRS.SYSLIB
// OPTION CATAL,DECK
// EXEC ASMA90,SIZE=(ASMA90,64K)
          .
          .
    member IJBEDEF.Z
          .
          .
/*
// EXEC LNKEDT,PARM='MSHP'
/&
* $$ EOJ
```

```
*********************************************************************
*       PANEL DATA DEFINITIONS                                      *
*********************************************************************
        IJBDEF PANEL,TSYST,,,'SYSTEM:','='
        IJBDEF PANEL,TSYSX,,,' ','='
        IJBDEF PANEL,TTITL,,,'      z/VSE 3.1     ','='
        IJBDEF PANEL,TUSER,,,'USER:','='
        IJBDEF PANEL,TUSEX,,,' ','='
        IJBDEF PANEL,TTIME,,,'TIME:','='
        IJBDEF PANEL,TTIMX,,,' ','='
        IJBDEF PANEL,TDISP,,,'TURBO','='
        IJBDEF PANEL,TDISPX,,,' ','='
        IJBDEF PANEL,TACMD,,,'==>','='
        IJBDEF PANEL,TCMD,,,' ','='
        IJBDEF PANEL,TDMSG,,,' ','='
        IJBDEF PANEL,TFILT,,,' ','='
        IJBDEF PANEL,TFILTR,,,'FILTER:','='
        IJBDEF PANEL,TACTM,,,'ACT_MSG:','='
        IJBDEF PANEL,TFILX,,,' ','='
        IJBDEF PANEL,THOLD,,,'HOLD','='
        IJBDEF PANEL,THRUN,,,'HOLDRUN','='
        IJBDEF PANEL,TNOHLD,,,'NOHOLD','='
        IJBDEF PANEL,TDIRC,,,' ','='
        IJBDEF PANEL,TPAUS,,,'PAUSE:','='
        IJBDEF PANEL,TPAUSX,,,' ','='
        IJBDEF PANEL,TSCRL,,,'SCROLL:','='
        IJBDEF PANEL,TSCRLX,,,' ','='
        IJBDEF PANEL,TIMSG,,,' ','='
        IJBDEF PANEL,TMESG,,,'MESSAGE','='
        IJBDEF PANEL,TSUSP,,,'SUSPEND','='
        IJBDEF PANEL,TMODE,,,'MODE:','='
        IJBDEF PANEL,TMODXC,,,'CONSOLE','='
        IJBDEF PANEL,TMODXCM,,,'CONSOLE ..MORE','='
        IJBDEF PANEL,TMODXCD,,,'CONSOLE ..HOLD','='
        IJBDEF PANEL,TMODXR,,,'REDISPLAY','='
        IJBDEF PANEL,TMODXE,,,'EXPLANATION','='
        IJBDEF PANEL,TMODXH,,,'HELP','='
*********************************************************************
*       PF KEY DEFINITIONS                                          *
*********************************************************************
        IJBDEF PFKEY,1,C,'1=HLP','=','%HELP'
        IJBDEF PFKEY,2,C,'2=CPY','=','%COPY ''?CL'
        IJBDEF PFKEY,3,C,'3=END','=','%END'
        IJBDEF PFKEY,4,C,'4=RTN','=','%RETURN'
        IJBDEF PFKEY,5,C,'5=DEL','=','%DELETE ''?CL'',''?IN'
        IJBDEF PFKEY,6,C,'6=DELS','=','%DELETE ''?CL'',''SYSTEM'
        IJBDEF PFKEY,7,C,'7=RED','=','%REDISPLAY ''?IN'
        IJBDEF PFKEY,8,C,'8=CONT','=','%CONTINUE'
        IJBDEF PFKEY,9,C,'9=EXPL','=','%EXPLAIN ''?TK'
        IJBDEF PFKEY,10,C,'10=HLD','=','%CHANGE ''HOLD'
        IJBDEF PFKEY,11,C,'       ','=',''
        IJBDEF PFKEY,12,C,'12=RTRV','=','%RETRIEVE'
        IJBDEF PFKEY,ENTER,C,'INPUT','=','?IN'
        IJBDEF PFKEY,CLEAR,C,'CLEAR','=','%CLEAR'
*
```

Figure 92. Contents of Member IJBEDEF.Z (Part 1 of 4)

```
         IJBDEF PFKEY,1,R,'1=HLP','=','%HELP'
         IJBDEF PFKEY,2,R,'2=CPY','=','%COPY ''?CL'
         IJBDEF PFKEY,3,R,'3=END','=','%REDISPLAY E'
         IJBDEF PFKEY,4,R,'     ','=',''
         IJBDEF PFKEY,5,R,'    ','=',''
         IJBDEF PFKEY,6,R,'6=CNCL','=','%REDI C'
         IJBDEF PFKEY,7,R,'7=BWD','=','%REDI ''?CL'';''B,''?IN'
         IJBDEF PFKEY,8,R,'8=FWD','=','%REDI ''?CL'';''F,''?IN'
         IJBDEF PFKEY,9,R,'9=EXPL','=','%EXPLAIN ''?TK'
         IJBDEF PFKEY,10,R,'10=INP','=','?IN'
         IJBDEF PFKEY,11,R,'     ','=',''
         IJBDEF PFKEY,12,R,'12=INFO','=','%CHANGE INFO'
         IJBDEF PFKEY,ENTER,R,'REDISPLY','=','%REDI ''?CL'';''?IN'
         IJBDEF PFKEY,CLEAR,R,'CLEAR','=','%CLEAR'
*
         IJBDEF PFKEY,1,E,'1=HLP','=','%HELP'
         IJBDEF PFKEY,2,E,'2=CPY','=','%COPY ''?CL'
         IJBDEF PFKEY,3,E,'3=END','=','%END'
         IJBDEF PFKEY,4,E,'     ','=',''
         IJBDEF PFKEY,5,E,'    ','=',''
         IJBDEF PFKEY,6,E,'     ','=',''
         IJBDEF PFKEY,7,E,'7=BWD','=','%BACKWARD'
         IJBDEF PFKEY,8,E,'8=FWD','=','%FORWARD'
         IJBDEF PFKEY,9,E,'9=EXPL','=','%EXPLAIN ''?TK'
         IJBDEF PFKEY,10,E,'10=INP','=','?IN'
         IJBDEF PFKEY,ENTER,E,'EXPLAIN','=','%EXPLAIN ''?TK'
         IJBDEF PFKEY,CLEAR,E,'CLEAR','=','%CLEAR'
*
         IJBDEF PFKEY,1,H,'1=HLP','=','%HELP'
         IJBDEF PFKEY,2,H,'    ','=',''
         IJBDEF PFKEY,3,H,'3=END','=','%END'
         IJBDEF PFKEY,4,H,'     ','=',''
         IJBDEF PFKEY,5,H,'    ','=',''
         IJBDEF PFKEY,6,H,'     ','=',''
         IJBDEF PFKEY,7,H,'7=BWD','=','%BACKWARD'
         IJBDEF PFKEY,8,H,'8=FWD','=','%FORWARD'
         IJBDEF PFKEY,9,H,'     ','=',''
         IJBDEF PFKEY,10,H,'10=INP','=','?IN'
         IJBDEF PFKEY,ENTER,H,'HELP','=','%HELP'
         IJBDEF PFKEY,CLEAR,H,'CLEAR','=','%CLEAR'
*
**********************************************************************
*        CONSOLE ROUTER LOCAL MESSAGE DEFINITIONS, RANGE M1 - M20    *
**********************************************************************
M1       IJBDEF MSG,'0D18I INVALID INPUT','='
M2       IJBDEF MSG,'0D14I COMMAND IGNORED','='
M3       IJBDEF MSG,'0D11I INVALID REPLY-ID','='
M4       IJBDEF MSG,'0D10I COMMAND/REPLY NOT AUTHORIZED','='
M5       IJBDEF MSG,'0D19I ATTENTION ROUTINE NOT ACTIVE','='
M6       IJBDEF MSG,'0D24I REDISPLAY PROCESSOR NOT ACTIVE','='
M7       IJBDEF MSG,'0D21I INPUT REJECTED BY EXTERNAL EXIT','='
```

Figure 92. Contents of Member IJBEDEF.Z (Part 2 of 4)

```
M8       IJBDEF MSG,'0D91I INPUT NOT ACCEPTED DUE TO REMOTE OPERATING M-
               ODE','='
M9       IJBDEF MSG,'0D92I REDISPLAY MODE ALREADY ACTIVE FOR OTHER USER-
               ','='
M10      IJBDEF MSG,'0D93I COMMAND NOT ACCEPTED','='
***********************************************************************
*        HARD COPY FILE LOCAL MESSAGE DEFINITIONS, RANGE M21 - M40   *
***********************************************************************
M21      IJBDEF MSG,'0D26E I/O ERROR ON HARD COPY FILE','='
M22      IJBDEF MSG,'0D29E INCORRECT LENGTH DURING I/O FOR HARD COPY FI-
               LE','='
M23      IJBDEF MSG,'0D51I EXTENT FAILED','='
M24      IJBDEF MSG,'0D52I GETVIS FAILED','='
M25      IJBDEF MSG,'0D56E INCONSISTENT STATE DURING HARD COPY FILE PRO-
               CESSING','='
M26      IJBDEF MSG,'0D80I INVALID REDISPLAY COMMAND','='
M27      IJBDEF MSG,'0D81I A TRAILING COMMA IS NOT VALID','='
M28      IJBDEF MSG,'0D82I FUNCTION HOLD AND A SUBFILTER ARE NOT COMPAT-
               IBLE','='
M29      IJBDEF MSG,'0D83I REDISPLAY COMMAND IS CANCELLED','='
M30      IJBDEF MSG,'0D84I REDISPLAY MODE IS TERMINATED','='
M31      IJBDEF MSG,'0D85I ACTION CANCEL DOES NOT ALLOW OTHER OPERANDS'-
               ,'='
M32      IJBDEF MSG,'0D86I NO REDISPLAY COMMAND/MODE IS ACTIVE, COMMAND-
                IGNORED','='
M33      IJBDEF MSG,'0D22I INSUFFICIENT GETVIS FOR REQUESTED FUNCTION',-
               '='
*
***********************************************************************
*        CONSOLE APPLICATION LOCAL MESSAGES, RANGE M41 - M80         *
***********************************************************************
M41      IJBDEF MSG,'0D61I PRESS CONTINUE TO RESUME','='
M42      IJBDEF MSG,'0D62I SCREEN IS FULL WITH HOLD MESSAGES (SET ACT_M-
               SG TO NOHOLD)','='
M43      IJBDEF MSG,'0D63I PF/PA KEY NOT DEFINED','='
M44      IJBDEF MSG,'0D64I COMMAND NOT ALLOWED IN THIS MODE','='
M45      IJBDEF MSG,'0D65I COMMAND NOT ALLOWED FROM THE INPUT LINE','='
M46      IJBDEF MSG,'0D66I INVALID CURSOR POSITION/LINE NUMBER FOR THIS-
                COMMAND','='
M47      IJBDEF MSG,'0D67I COMMAND INVALID','='
M48      IJBDEF MSG,'0D68I OPERAND INVALID','='
M49      IJBDEF MSG,'0D69I PRESS END TO RESUME','='
M50      IJBDEF MSG,'0D70I NO MORE EXPLAIN/HELP DATA AVAILABLE','='
M51      IJBDEF MSG,'0D71I NO EXPLAIN/HELP DATA FOUND','='
M52      IJBDEF MSG,'0D72I TRY AGAIN LATER','='
M53      IJBDEF MSG,'0D73I CONSOLE DEACTIVATED, HIT ENTER TO RESUME','=-
               '
M54      IJBDEF MSG,'0D74I EXPLAIN FILE ACCESS FAILURE','='
M55      IJBDEF MSG,'0D75I EXPLAIN SUPPORT NOT ACTIVE','='
M56      IJBDEF MSG,'0D76I EXPANSION FAILURE','='
M57      IJBDEF MSG,'0D77I DICTIONARY COULD NOT BE LOADED','='
*
```

Figure 92. Contents of Member IJBEDEF.Z (Part 3 of 4)

```
*********************************************************************
*       CONSOLE PARAMETER SETTINGS                                  *
*********************************************************************
*
        IJBDEF DEFAULT,HOLD,RUN     (YES/RUN/NO)  DEFAULT=RUN  PN78356
        IJBDEF DEFAULT,ALARM,YES    (YES/NO)            DEFAULT=YES
        IJBDEF DEFAULT,INFO,NONE    (NONE/TSTAMP/USERID) DEFAULT=NONE
        IJBDEF DEFAULT,PAUSE,1      (00 GE NN LE 99)    DEFAULT=1
        IJBDEF DEFAULT,SCROLL,1     (0 GE N LE 9)       DEFAULT=1
*
*********************************************************************
*       GENERATE THE TABLES                                         *
*********************************************************************
        IJBDEF GEN,$IJBEDEF
```

Figure 92. Contents of Member IJBEDEF.Z (Part 4 of 4)

End of Programming Interface information

Chapter 20. ZONE Specifications and Daylight Saving Time

With the *Tailor IPL Procedure* dialog, you can add or modify zone specifications to define time zones, and to be able to switch between standard and daylight saving time (summertime) without changing the IPL startup procedure each time.

To access the dialog, start with the *z/VSE Function Selection* panel and select:
2 (Resource Definition)
4 (Hardware Configuration and IPL)
3 (Tailor IPL Procedure)

Administrator Fast Path: 243	Synonym Default: ____ Yours:

You get a list of the IPL parameters that can be modified. Select the ZONE parameter by entering 1 next to it.

```
TAS$MAS2                  TAILOR IPL PROCEDURE

Enter the required data and press ENTER.

                                                   IPL procedure = $IPLESA

To modify one or more of the following IPL parameters, place a 1 next to it.

     _     Supervisor  Modify console, supervisor- and storage option
     _     SYS         Modify SYS command parameters
     _     UNATTENDED  Modify unattended node parameters
     _     DLA         Modify label area definition
     _     DPD         Modify page data set definition
     _     DLF         Modify lock file definition
     _     DEF         Modify recorder file and catalog assignment
     1     ZONE        Modify ZONE specifications
     _     APPC/VM     Modify VSE APPC/VM specification
     _     SVA         Modify shared virtual area definition


PF1=HELP      2=REDISPLAY  3=END                    5=PROCESS
```

Figure 93. Tailor IPL Procedure Dialog

On the *ZONE SPECIFICATION* panel, you can define the zone direction and the zone hours, or you can specify a zone id. These two possibilities define static values in your IPL procedure.

```
TAS$ICM6          TAILOR IPL PROCEDURE: ZONE SPECIFICATION

 Enter the required data and press ENTER.



ZONE DIRECTION............... _           Direction to you from Greenwich
                                          England (1 = East   2 = West)

ZONE HOURS................... __          Hours to you from Greenwich, England
                                          (Two digits between 00 and 23)

ZONE ID...................... CES         Time Zone Definition.






 PF1=HELP      2=REDISPLAY  3=END                    5=PROCESS
               8=FORWARD
```

Figure 94. Panel for Modifying Zone Specifications

Note

If you want to use the possibility to switch between times, make sure that you have no static ZONE definition specified on the panel shown above (TAS$ICM6). The switching will be ignored when the IPL procedure contains a static SET ZONE definition.

To add or change the Time Zone Definition (ZONE ID), delete the values shown on the ZONE DIRECTION and ZONE HOURS lines and press PF8 to define the characteristics of the ZONDEF specifications (see Figure 95 on page 347). If your definitions are already completed on this panel, press PF5 to process the entered values.

Note that if you do not want to have a SET ZONE command in your IPL procedure, (for example, if you are running under VM), you need to delete all the values entered on panel TAS$ICM6.

ZONEDEF Specification

Use the *Zonedef Specification* dialog to define system time zones according to their difference from Greenwich Mean Time (GMT).

```
TAS$ICMB        TAILOR IPL PROCEDURE: ZONEDEF SPECIFICATION

Enter the required data and press ENTER.

OPTIONS: 1 = ADD   2 = ALTER    5 = DELETE

 OPT       ZONE ID       ZONE DIRECTION    ZONE HOURS

  _         CES            1                 00
  _         CET            1                 01
  _         EST            2                 05
  _         CST            2                 06
  _         EDT            2                 04
  _         CDT            2                 05
  _         ___            _                 __
  _         ___            _                 __
  _         ___            _                 __
  _         ___            _                 __


PF1=HELP       2=REDISPLAY  3=END                    5=PROCESS
PF7=BACKWARD   8=FORWARD
```

Figure 95. ZONEDEF Specification Panel

The following values can be specified:

ZONE ID
Enter a three character name for this ZONE definition. You can select any name you want. It refers to a specific zone value. You can define up to 10 new ZONE IDs. The examples shown, are the names of the official time zones division. For example:
- CES for Central European Summertime
- CET for Central European Standard Time
- EST for Eastern Standard Time
- EDT for Eastern Daylight Saving Time
- CST for Central Standard Time
- CDT for Central Daylight Saving Time

ZONE DIRECTION
Enter the direction to you from Greenwich, England. 1 = East, 2 = West. For example, define 2 for USA which is to the west of Greenwich.

ZONE HOURS
Enter the hours to you from Greenwich, England. Enter two digits between 00 and 23. Only zone hours, no zone minutes are supported.

Press PF8 to get the *ZONEBDY Specification* panel, or press PF5 to process the entered values.

ZONEBDY Specification

```
TAS$ICMC        TAILOR IPL PROCEDURE: ZONEBDY SPECIFICATION

Enter the required data and press ENTER.

OPTIONS: 1 = ADD              5 = DELETE

 OPT      BEGIN DATE       BEGIN TIME    ZONE ID
          mmddyyyy         hhmmss

  _       04012004         000001          CES
  _       10012004         000001          CET
  _       04012005         000001          CES
  _       10012005         000001          CET
  _       ________         ______          ___
  _       ________         ______          ___
  _       ________         ______          ___
  _       ________         ______          ___
  _       ________         ______          ___
  _       ________         ______          ___

PF1=HELP      2=REDISPLAY  3=END                    5=PROCESS
PF7=BACKWARD
```

Figure 96. ZONEBDY Specification Panel

With the *ZONEBDY Specification* panel you can define the date and time when z/VSE should begin to use a given time zone. Usually, you use these definitions to switch between standard and daylight saving local times.

BEGIN DATE
Enter the date, in the format mmddyyyy, on which z/VSE should begin using a given time zone. You can define up to 20 dates.

BEGIN TIME
Enter the local time in the format hhmmss, on which z/VSE should begin using a given time zone.

ZONE ID
Enter a three character time zone definition you specified before on the *ZONEDEF Specification* panel.

Press PF5 to process the data, or press PF7 to re-display the entered values.

Note that you have to IPL the system in order to switch to the new time zone.

The statements created by the dialog are documented, together with examples, in the manual *z/VSE System Control Statements* under “SET ZONEDEF” and “SET ZONEBDY”.

Chapter 21. Displaying System Status and Storage Information

Dialogs Available

z/VSE provides three dialogs for system status and storage display:

- Display System Activity
- Display Channel and Device Activity
- Display CICS TS Storage
- Display Storage Layout

Display System Activity or Channel and Device Activity

The two dialogs, *Display System Activity* and *Display Channel and Device Activity*, provide system status information for daily operation. For this reason, these two dialogs are described in detail in the manual *z/VSE Operation* under "Displaying System Activity" and under "Displaying Channel and Device Activity".

The system administrator can access these dialogs from the *z/VSE Function Selection Panel* as follows:

1. The *Display System Activity* dialog with fast path **361**.
2. The *Display Channel and Device Activity* dialog with fast path **362**.

The display of the first dialog is updated automatically in intervals. This interval, 10 seconds for example, can be changed by the system administrator. How to change the interval time for this dialog is described under "Changing the Dialog Interval Time" on page 355.

For the second dialog, press ENTER to get an updated display.

Display CICS TS Storage

For tuning and debugging purposes, a detailed display of the CICS TS partition layout may be required. z/VSE provides such a display via fast path 364. The display provides detailed information about storage allocation and usage as shown in Figure 97 on page 350. The help text (PF1) provides a description of the information shown.

For a detailed discussion of CICS TS virtual storage refer to the *CICS Performance Guide*.

Instead of using the dialog, it may be sometimes more convenient entering the command **IEDC** on the CICS TS command line to get the same display.

```
IESADMDCST                    DISPLAY CICS TS STORAGE        Time:  09:40:19
  Applid: DBDCCICS    Sysid: CIC1    Jobname: CICSICCF    CICS TS Level:   111

Storage Protection ...... INACTIVE       Reentrant Programs ..... PROTECT
                                         CICS Trace Table size..      80
Extended DSA:                      (All sizes in kbyte)    LIMIT   25600
                                   ECDSA   EUDSA   ESDSA   ERDSA  Totals
  Current DSA Size ............     2048    1024    1024    6144   10240
  Current DSA used ............     1876      64       8    5220    7168
 *Peak DSA used ...............     1884      64       8    5220
  Peak DSA Size ...............     2048    1024    1024    6144   10240
  Largest free area/Free Storage    0.95    1.00    1.00    0.94
  Times short-on-storage (SOS)..       0       0       0       0       0

DSA:                                                       LIMIT    5120
                                    CDSA    UDSA    SDSA    RDSA  Totals
  Current DSA Size .............     512     256     512     512    1792
  Current DSA used .............     344       8     456     344    1152
 *Peak DSA used ................     352      28     456     344
  Peak DSA Size ................     512     256     512     512    1792
  Largest free area/Free Storage.   0.88    1.00    0.86    0.86
  Times short-on-storage (SOS)...      0       0       0       0       0
PF1=HELP      2=REFRESH   3=END       4=RETURN
```

Figure 97. Display CICS TS Storage Dialog

Using the Display Storage Layout Dialog

The data displayed by the *Display Storage Layout* dialog helps the system administrator optimize storage layout, particularly the partition layout. This is of interest especially for partitions which host permanently running programs, such as VSE/POWER or VTAM or applications installed by the user.

The values displayed may indicate that changes like the following need to be considered:

- A reduction or increase of the partition size.
- A reduction or increase of the GETVIS area of a partition.

Accessing the Dialog

To access the *Display Storage Layout* dialog, start with the *z/VSE Function Selection Panel* and select:

3 (Operation)
6 (System Status)
3 (Display Storage Layout)

Administrator	Synonym
Fast Path: 363	Default: ____ Yours:

Figure 98 on page 351 shows the initial display.

The panel is divided into a left part for static partitions and a right part for dynamic classes.

```
IESADMDSS1                    DISPLAY STORAGE LAYOUT

-------------------------------------------------------------------------------
|                 SVA (31 bit):       14M                                     |
-------------------------------------------------------------------------------| Data
|       X  X  X       X  X            |                                       | Space:
| 16M - - - - - - - - - - - - - - - - | - - - - - - - - - - - - - - - -       | 9440K
|       X  X  X       X  X            |                                       |
| X     X  X  X       X  X            |                                       |
| X     X  X  X       X  X            |                                       |
| X  X  X  X  X  X  X  X  X  X  X  X  |                                       |
| BG F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB | C  J  P  R  S  Y  Z                   |
| 0  1  2  3  4  5  6  7  8  9  A  B  |                                       |
-------------------------------------------------------------------------------|
|            |     SHARED PARTITIONS: none                                    | Avail-
|      5120K |     SVA (24 bit):      3652K        Unused:  704K              | able:
|            |     SUPERVISOR:         764K                                   |   70M
-------------------------------------------------------------------------------
Private space size:      70M                   Total virtual storage:  264M
PARTITION ID / DYNAMIC CLASS: __   Specify for more information

PF1=HELP      2=REFRESH   3=END       4=RETURN              6=SVA
```

Figure 98. Dialog Entry Panel

Address space IDs connected by **--** indicate that the corresponding partitions share their address space.

The 16M line symbolically marks the transition to 31-bit addresses (only shown for systems with an address space > 16MB). An **X** above the 16M line indicates that space beyond 16M has been allocated for the corresponding partition. A **|** indicates that this space is used by another static partition in the same address space. The address space IDs are shown below the partition IDs.

The size of the private area in each address space is shown at the bottom of the display.

The allocations of the dynamic classes are shown on the right part with the corresponding dynamic class IDs.

The panel allows you to specify the ID of a static partition or the class of a dynamic partition or even a dynamic partition ID directly you want to display. After pressing ENTER, either the storage values of the partition selected are shown together with the name of the job currently running in the partition or a table with the active dynamic partitions of this class. Pressing PF6 on any panel gives you an additional display showing the storage values for the SVA (shared virtual area).

Static Partition Layout Panel

Figure 99 on page 352 shows the static partition layout for an F8 (CICS2) partition. You get this display by entering F8 in the initial panel shown in Figure 98.

```
IESADMDSPL                STATIC PARTITION LAYOUT

          PARTITION: F8                         JOB NAME: CICS2
                                                PHASE:    DFHSIP
-------------------------------------------------------------------------------
'035FFFFF'X                             | GETVIS ANY             |
                                        | USED:              33M |
                                        | FREE:              17M |
- -(16M)- - - - - - - - - - - - - - - - | HIGH WATER MARK:   40M |
          A GETVIS BELOW                | LARGEST FREE BLOCK: 17M |
          | USED:               5636K   |                        |
          | FREE:               6648K   |                        |    50M
          | HIGH WATER MARK:      12M   |                        |
'00401000'X | LARGEST FREE BLOCK: 6644K  V                       |
-------------------------------------------------------------------------------
            PROGRAM AREA (EXEC SIZE)     LOADED PHASE:      582B |
'00400000'X                              AVAILABLE:        3514B | 4096B
-------------------------------------------------------------------------------

                                                     PARTITION:   50M

PF1=HELP      2=REFRESH   3=END       4=RETURN              6=SVA
```

Figure 99. Static Partition Layout Panel

The panel is divided into the GETVIS part and the program part. The left part of the GETVIS part shows GETVIS BELOW (<16M) information, whereas the right part shows information about GETVIS ANY.

Notes:

1. The GETVIS ANY area may reach into the GETVIS BELOW area, that means, the USED area (31 bit) and the FREE area (31 bit) include the corresponding 24-bit areas, whereas the largest free block can include the 24-bit area.
2. Any GETVIS area not initialized appears as * on the panel together with a message.
3. High Water Mark is the largest partition GETVIS size used in the current job step.

The addresses shown in Figure 99 have the following meaning:

```
035FFFFF = end address of partition
00600000 = start address of partition GETVIS
00400000 = start address of partition
```

You may enter the ID of any static or dynamic partition directly on this panel. You can also enter the ID of a dynamic class to get the list of the active dynamic partitions of that class displayed.

Dynamic Partition Layout Panel

Figure 100 on page 353 shows the layout of dynamic partition Z1 which is used as example. You get this display by entering in the initial panel (Figure 98 on page 351) class Z and selecting on the subsequent panel Z1.

```
IESADMDDPL              DYNAMIC PARTITION LAYOUT
     PARTITION: Z1        JOB NAME: PAUSEF2        PHASE: LIBR
------------------------------------------------------------------------------
                                         | GETVIS ANY              |
                                         | USED:               0   |
                                         | FREE:               0   |
- - - - - - - - - - - - - - - - - - - - -| HIGH WATER MARK:    0   |
'008FFFFF'X A GETVIS BELOW               | LARGEST FREE BLOCK:  0  |
            | USED:               32K  |                           |
            | FREE:             3936K  |                           | 3968K
            | HIGH WATER MARK:    32K  |                           |
'00520000'X | LARGEST FREE BLOCK: 3936K  V                         |
------------------------------------------------------------------------------
              PROGRAM AREA (EXEC SIZE)     LOADED PHASE:        80K |
'00420000'X                                AVAILABLE:          944K | 1024K
------------------------------------------------------------------------------
              DYNAMIC SPACE GETVIS         USED:                24K |
                                           FREE:               104K |  128K
'00400000'X                                HIGH WATER MARK:     40K |
------------------------------------------------------------------------------
                                                             PARTITION: 5120K
PF1=HELP      2=REFRESH   3=END       4=RETURN               6=SVA
```

Figure 100. Dynamic Partition Layout Panel

The GETVIS information shown is the same as for static partitions. In addition, the dynamic space GETVIS area is shown. The meaning of the addresses shown is as follows:

```
009FFFFF = end address of partition
00620000 = start address of partition GETVIS
00520000 = start address of partition
00500000 = start address of dynamic space GETVIS
```

Notes:

1. The value 0 in GETVIS ANY means that the total GETVIS area is below 16MB.
2. High Water Mark is the largest dynamic space GETVIS area size used in the current VSE/POWER job.

If the value for FREE is frequently very low you should consider a larger size for the dynamic space GETVIS area by decreasing the partition size. If the value for FREE is frequently very high consider an increase of the partition size. For additional information, you may also refer to the chapter “Storage Management” in the manual *z/VSE Guide to System Functions*.

The panel allows you to specify for display the ID of a static partition, or the class of a dynamic partition, or a dynamic partition ID.

SVA Layout Panel

By pressing PF6 on one of the previous panels you get the SVA layout of your system; Figure 101 shows an example.

By pressing PF6 on this panel, you get back to the partition layout panel.

```
IESADMDSV1              SHARED VIRTUAL AREA LAYOUT

-------------------------------------------------------------------------------
'043FFFFF'X SYSTEM GETVIS(31) USED:   (HWM: 2072K ) |  1224K |        |
'03CEE000'X                   FREE:                 |  6016K |  7240K |
--------------------------------------------------------------------|    14M
            PROGRAM AREA(31)  USED:                 |  3002K |        |
'03600000'X                   AVAILABLE:            |  4094K |  7096K |
===============================================================================
'003EFFFF'X V-POOL                                  |    64K |        |
'003C5000'X SYSTEM LABEL AREA (SLA)                 |   108K |   172K |
--------------------------------------------------------------------|
            SYSTEM GETVIS(24) USED:   (HWM:  848K ) |   836K |        |
'00249000'X                   FREE:                 |   684K |  1520K |
--------------------------------------------------------------------|  3372K
            PROGRAM AREA(24)  USED:                 |  1365K |        |
'000B5000'X                   AVAILABLE:            |   251K |        |
-----------------------------------------------------------|  1680K |
            SYSTEM DIRECTORY LIST (SDL)             |    64K |        |
'000A5000'X                                         |        |        |
-------------------------------------------------------------------------------
PF1=HELP      2=REFRESH   3=END       4=RETURN               6=PARTITION
```

Figure 101. SVA Layout Panel

High Water Mark (HWM) is the largest size used since the last IPL. The meaning of the SVA addresses and size values is as follows:

ADDRESSES:

```
023FFFFF = End address of SVA
0230B000 = Start address of system GETVIS area (31 bit)
02300000 = Start address of system program area (31 bit)
003FFFFF = End address of virtual pool (V-POOL)
003C5000 = Start address of system label area (SLA)
00255000 = Start address of system GETVIS area (24 bit)
008F0000 = Start address of system program area (24 bit)
00087000 = Start address of SVA (system directory list, SDL)
```

GETVIS SIZE VALUES:

```
 980K   = Size of the system GETVIS area 31 bit (used and free,
          including GETVIS control area).
1472K   = Size of the system GETVIS area 24 bit (used and free,
          including GETVIS control area).
```

SVA SIZE VALUES:

```
 128K     = Size of the virtual pool area (V-POOL)
 108K     = Size of the system label area (SLA)
  32K     = Size of the system directory list (SDL)
  44K     = Size of the system program area (31 bit, used and free)
 236K     = Sum of the size of the virtual pool area (V-POOL) and
            size of the system label area (SLA)
1848K     = Size of the system program area (24 bit, used and free,
            plus the size of the system directory list, SDL)
```

In the rightmost column the display shows;

```
1024K     = Size of SVA (31 bit)
3556K     = Size of SVA (24 bit)
```

Changing the Dialog Interval Time

The *Display System Activity* dialog automatically redisplays current system activity every 15 seconds. You can change the interval time or have the display updated only when the user presses ENTER. The information below outlines what you must do to make such changes.

1. Use the *Maintain Application Profiles* dialog and locate the application profile **IESLA**.
2. Create a new application profile. Use profile IESLA as a model. You cannot change the IESLA application profile itself.
3. In the DATA field, enter the value you want for the time interval:

   0 Use ENTER key to refresh the display.

   10-59 Number of seconds for time interval.

   If you do not enter a value or you enter an incorrect one, the dialog does not display an error message. It uses 15 as the default.
4. You must include the new profile either in a user profile, a selection panel, or both. You can use the following dialogs:
   - *Maintain User Profiles*
   - *Maintain Selection Panels*

The dialog calculates information using VSE job accounting tables and other system control blocks. Therefore, VSE job accounting must be active in the system (SYS JA=YES at IPL time). If job accounting is not active, you cannot access the dialog. In this case, if you try to access the dialog, the system displays an information message on the selection panel.

All jobs should include // JOB and /& statements for correct job accounting and a more accurate display of system activity. This includes startup jobs and jobs that are initiated during startup.

Chapter 22. Collecting Additional z/VSE Activity Data

This enhancement takes measurements of the **system** and **channel/device** activity of a z/VSE system. It stores the resulting data in temporary storage queues of the CICS Transaction Server (CICS TS) and activates a user exit program for further processing and analysis of the data. Note that the data provided is almost identical to the data collected by the *Display System Activity* and the *Display Channel and Device Activity* dialogs. Both dialogs are described in detail in the *z/VSE Operation* manual. You are recommended to familiarize yourself with these dialogs before working with the enhancements described in this chapter.

The advantage of the enhanced support is that activity data can be collected over a longer period of time, and can be saved for later analysis. Compared to the dialogs, the support is terminal-independent, because it communicates with the user exit program only.

Notes:

1. The enhancement applies to data collection, but not to the interpretation of the data collected. It is **not** a replacement for any kind of performance tool.
2. The activity measurements require the Job Accounting parameter in the IPL SYS command to be set (JA=YES). This is the z/VSE default in a newly installed system.

The enhanced support includes CICS TS transactions, programs, queues, and a skeleton as follows:

IEXM
This is the main transaction used to start a measurement cycle. IEXM requires measurement parameters as input.

IEXA
IEXM invokes this transaction if **system activity** is to be measured. IEXA stores the measurement data in the temporary storage queue IESAIEXA and activates a user exit program; either IESDALOG, which is the default, or the one specified as input parameter for IEXM.

IEXS
IEXM invokes this transaction if **channel/device activity** is to be measured. IEXS stores the measurement data in the temporary storage queue IEDSIEXS and activates a user exit program; either IESCHLOG, which is the default, or the one specified as input parameter for IEXM.

IESAIEXA
This is the CICS TS temporary storage queue in which transaction IEXA saves the current system activity data of a single measurement sample for further processing by a user exit program. The data is overwritten when the next sample is taken.

IEDSIEXS
This is the CICS TS temporary storage queue in which transaction IEXS saves the current channel/device activity data of a single measurement sample for further processing by a user exit program. The data is overwritten when the next sample is taken.

IESDAOUT
This is the default name of the **temporary storage queue** in which the sample user exit program in skeleton SKEXITDA saves the contents of the temporary storage queue IESAIEXA (which contains system activity data of a single measurement sample only).

IESCHOUT This is the default name of the **temporary storage queue** into which the sample user exit program in skeleton SKEXITDA saves the contents of the temporary storage queue IEDSIEXS (which contains channel/device activity data of a single measurement sample only).

IESDALOG IESDALOG is the default name of the **user exit program** to be activated in case of system activity measurements.

IESCHLOG IESCHLOG is the default name of the **user exit program** to be activated in case of channel/device activity measurements.

SKEXITDA This skeleton provides a sample user exit program for saving measurement data. You can use the skeleton to modify the user exit program provided to meet the specific needs of your environment. The skeleton is available in VSE/ICCF library 59.

IESX This is the CICS TS abend code, which is set if a link to a user exit program is not possible.

Taking Measurements

Main transaction IEXM is used to start a measurement cycle and requires the following input parameters:
- Name of user exit program (EXIT)
- Activity to be measured (MEASURE)
- Device range for channel/device (DEVRANGE)
- Start time for measurement (STARTTIME)
- Stop time for measurement (STOPTIME)
- Interval for measurement (INTERVAL)
- Cancel request (optional) (FORCE)

To start a measurement cycle, you must call transaction IEXM and provide the input parameters needed for the measurement. You can enter the information string at the system console or at any other console of your z/VSE system. Access is also possible from a native CICS TS subsystem. An example is given below:

```
IEXM E(IESDALOG) M(S) STA(070000) STO(073000) I(000100)
```

The input requests transaction IEXM to initiate a measurement cycle for system activity starting at 7:00 and ending at 7:30 and take these measurements in intervals of 60 seconds. The example uses the short forms of the input parameters which are described in detail below.

Format of Input Parameters for Transaction IEXM

An input parameter consists of a keyword plus its value surrounded in parentheses. If the value is not specified, the defaults become effective. Parameters are separated by one or more blanks. The maximum length allowed for the string of input parameters is 100 characters.

If an error occurs in the parameter specifications, the user exit program is notified and no measurement transaction is activated.

Following is a description of the keywords allowed and the values that can be specified for them. The characters shown in uppercase are required, those in lowercase are optional:

Exit(*cccccccc*)
This parameter defines the name of the user exit program invoked by transactions IEXM, IEXA, and IEXS.

The default name is IESDALOG for measuring system activity, and IESCHLOG for measuring channel and device activity.

Measure(Sa | Cda | All)
This parameter defines the type of measurement to be taken:
Sa – System Activity
Cda – Channel and Device Activity
All – Both activities

The default is "All".

Devrange(*ccc[-ccc]*)
This parameter is only meaningful if Channel and Device activity is to be measured. The maximum range is from 000 to FFF, which is also the default.

STArttime(*hhmmss*)
This parameter defines the measurement starting time. Allowed ranges are:
- hh – from 00 to 99

  A value greater 23 specifies a time on a day following the current one.
- mm – from 00 to 59
- ss – from 00 to 59

Note that the CICS TS rules for expiration times apply here. This means that if STArttime is smaller than the current time, two cases are possible:
- If STArttime + 6 hours is greater than the current time, the task is started immediately.
- Otherwise the task is started on the next day.

The default is to start measurements immediately.

Example:

If the STArtime is specified as 100000 (10am) and the current time is 5pm (7 hours greater), the task will be started the next day. With the same STArtime and a current time of 3pm, the task would be started immediately.

STOptime(*hhmmss*)
This parameter defines the stop time of measurement. Allowed ranges are:
- hh – from 00 to 99

  A value greater 23 specifies a time on a day following the current one.
- mm – from 00 to 59
- ss – from 00 to 59

The STOptime has to be larger than the STArttime. A STOptime on a day following the STArtime day is possible with one exception: the crossing of a year boundary is not allowed.

If the STArttime given causes the task to be started on the next day, 24 hours are added to the given STOptime automatically. If the STOptime is smaller than the current time, measurement is not started.

The default is not to stop measurements, which means that the task may be stopped by the user exit program or the operator, for example.

Interval(*hhmmss*)
This parameter defines the interval from one measurement sample to the next. Allowed ranges are:
- hh – from 00 to 99
- mm – from 00 to 59
- ss – from 00 to 59

The lower limit is 10 seconds; the default is 15 seconds.

Force(Yes | <u>No</u>)
This parameter allows you to cancel a running measurement to enforce, for example, the start of a new measurement.
- Yes – Cancel running measurement and start new measurement.
- No – Start new measurement only if no previously defined measurement is still scheduled. This is also the default.

Transactions IEXA and IEXS

Transactions IEXA and IEXS are called by transaction IEXM to perform the actual activity measurements.

IEXM invokes IEXA for system activity measurements, IEXS for channel and device activity measurements. Both transactions store the measurement data in a CICS TS temporary storage queue, either in IESAIEXA or in IEDSIEXS, and activate the corresponding user exit program.

The transactions activate the user exit program each time a measurement sample is taken and written to the CICS TS temporary storage. The exit program should save all or important performance data stored in IESAIEXA or IEDSIEXS since the data is overwritten when the next measurement sample is taken.

The user exit program specifies a return code for the transaction by which it was invoked. This return code indicates either to continue measurements or to terminate them, even if ending time has not been reached yet. When the specified ending time is reached, the exit program is invoked a last time to signal end of measurements.

User Exit Description

Note: User-written exit programs with a name other than IESDALOG or IESCHLOG must be added to the CICS TS table PPT as follows:

```
DFHPPT TYPE=ENTRY,PROGRAM=name,RSL=PUBLIC
```

User Exit Linkage Definition

When transactions IEXM, IEXA, or IEXS establish linkage to a user exit program, they provide linkage information in the CICS TS COMMAREA. The layout of the COMMAREA used to link to a user exit program is as follows:

```
Name      Length  Type    Contents
____________________________________________________________________________

UXPL       0      B       Introducer
UXPLNAME   8      C       'IESUXPL ', name of area
UXPLLENG   2      B       Length of area
UXPLEC     1      B       Event code, event which invoked exit program
  UXECFITI                X'01' first time invocation
  UXECLATI                X'02' last time invocation
  UXECSYST                X'11' measurement data of system activity
  UXECCHDE                X'21' measurement data of channels and devices
UXPLRC     1      B       Return code of the exit routine
                          X'00' ok - continue with measurement
                             anything else - stop measurements
UXPLERRC   1      B       Error code
                          X'00' ok - no error to report
                             anything else - refer to

Figure 107 on page 365
UXPLERRI   8      C       Further error information
UXPLSTAR   6      C       Input parameter: start time    (hhmmss)
UXPLSTOP   6      C       Input parameter: stop time     (hhmmss)
UXPLINTE   6      C       Input parameter: interval      (hhmmss)
UXPLDEVR   7      C       Input parameter: device range  (ccc-ccc)
UXPLTSQU   8      C       Name of TS-Queue containing measurement data
                          (IESAIEXA or IEDSIEXS)
```

Figure 102. COMMAREA Layout for Linkage to User Exit Program

Sample User Exit Program Provided by Skeleton SKEXITDA

Skeleton SKEXITDA in VSE/ICCF library 59 provides a sample user exit program for both types of measurement data: **system activity** and **channel/device activity**. The primary purpose of the program is to save the data collected before a new measurement takes place. You can use SKEXITDA to tailor the exit program according to the needs of your installation.

You may activate the user exit program provided to see how it works. Proceed as follows:

1. Copy skeleton SKEXITDA from VSE/ICCF library 59 to your primary VSE/ICCF library.
2. Rename SKEXITDA to IESDALOG.
3. Compile the sample program (without modifying it) by using the compile option (8) of the *Program Development Library* dialog for online assembler programs.

   z/VSE creates a phase named IESDALOG.PHASE and catalogs it into the sublibrary specified in compile skeleton C$$ASONL. This skeleton is stored in VSE/ICCF library 2. The sublibrary should be defined in the active LIBDEF PHASE chain for the CICSICCF partition (as shipped, this is partition F2).
4. Activate the new CICS TS phase by calling transaction CEMT as follows:

   CEMT SET PROG(IESDALOG) NEW
5. Invoke transaction IEXM as shown under "Taking Measurements" on page 358.

   For channel and device activity data you must specify EXIT(IESDALOG) as for system activity. This is because IESDALOG, as provided in skeleton

SKEXITDA, can handle both: system activity and channel/device activity data. If you do not specify IESDALOG explicitly, IEXM tries to invoke the default user exit program IESCHLOG.

Transactions IEXA and IEXS save the measurement data of a **single measurement** sample in a temporary storage queue. The name of this queue is IESAIEXA for system activity data, and IEDSIEXS for channel and device activity data. The temporary storage queues IESDAOUT and IESCHOUT are used to save the CICS TS COMMAREA and the measurement records of a **complete measurement cycle**. Therefore, program IESDALOG moves the data of a single measurement sample from IESAIEXA into IESDAOUT and from IEDSIEXS into IESCHOUT.

Program IESDALOG communicates with IEXM and IEXA via the CICS TS COMMAREA by receiving event codes and possibly error codes. IESDALOG responds by providing a return code.

When modifying the user exit program, it is helpful to analyze the content of IESDAOUT and IESCHOUT by using the CICS TS transaction CEBR. Examples of CEBR displays for IESDAOUT and IESCHOUT are shown in Figure 103 through Figure 106 on page 364. Figure 103 shows a sample CEBR IESDAOUT display in character representation:

```
 CEBR            TS QUEUE  IESDAOUT RECORD     1 OF    10    COL     1 OF  2052
 ENTER COMMAND ===>
      **************************  TOP OF QUEUE  *******************************
00001 IESUXPL .....             143740000015000-FFFIESAIEXA
00002 IESUXPL .....       .........................IESAIEXA
00003 IESAIEXA01/12/0014:37:12...3.......................................r....
00004 ....... C.................................. .... ..........C.......... P
00005 ....... Y100DITESYSADITTO   ............... .... .....N................
00006 IESUXPL .....       .........................IESAIEXA
00007 IESAIEXA01/12/0014:37:27...........................................r....
00008 ....... C.................................. .... ..........C.......... P
00009 ....... Y100DITESYSADITTO   ............... .... .....N................
00010 IESUXPL .....       .........................IESAIEXA
      *************************  BOTTOM OF QUEUE  *****************************





PF1 : HELP               PF2 : SWITCH HEX/CHAR     PF3 : TERMINATE BROWSE
PF4 : VIEW TOP           PF5 : VIEW BOTTOM         PF6 : REPEAT LAST FIND
PF7 : SCROLL BACK HALF    PF8 : SCROLL FORWARD HALF PF9 : VIEW RIGHT
PF10: SCROLL BACK FULL    PF11: SCROLL FORWARD FULL PF12: UNDEFINED
```

Figure 103. IESDAOUT Display in Character Representation

Figure 104 on page 363 shows a sample CEBR IESDAOUT display in hexadecimal representation:

```
 CEBR           TS QUEUE  IESDAOUT RECORD     1 OF    10    COL     1 OF  2052
 ENTER COMMAND ===>
      *************************  TOP OF QUEUE  *******************************
00001 C9C5E2E4E7D7D3400036010000404040404040404040404040404040F1F4F3F7F4F0F0F0
00002 C9C5E2E4E7D7D34000361100004040404040404040000000000000000000000000000000
00003 C9C5E2C1C9C5E7C1F0F161F1F261F9F6F1F47AF3F77AF1F2AC42CBF359B2BA0100000000
00004 0004000000000040C3FF0000000000000000000000000000000000000000000000000000
00005 0001000000000040E8F1F0F0C4C9E3C5E2E8E2C1C4C9E3E3D64040400000000000000000
00006 C9C5E2E4E7D7D34000361100004040404040404040000000000000000000000000000000
00007 C9C5E2C1C9C5E7C1F0F161F1F261F9F6F1F47AF3F77AF2F7AC42CC01AFFA3E000000000F
00008 0004000000000040C3FF0000000000000000000000000000000000000000000000000000
00009 0001000000000040E8F1F0F0C4C9E3C5E2E8E2C1C4C9E3E3D64040400000000000000000
00010 C9C5E2E4E7D7D34000360200004040404040404040000000000000000000000000000000
      *************************  BOTTOM OF QUEUE  *****************************




PF1 : HELP                PF2 : SWITCH HEX/CHAR     PF3 : TERMINATE BROWSE
PF4 : VIEW TOP            PF5 : VIEW BOTTOM         PF6 : REPEAT LAST FIND
PF7 : SCROLL BACK HALF    PF8 : SCROLL FORWARD HALF PF9 : VIEW RIGHT
PF10: SCROLL BACK FULL    PF11: SCROLL FORWARD FULL PF12: UNDEFINED
```

Figure 104. IESDAOUT Display in Hexadecimal Representation

Figure 105 shows a sample CEBR IESCHOUT display in character representation:

```
 CEBR           TS QUEUE  IESCHOUT RECORD     1 OF    12    COL     1 OF   348
 ENTER COMMAND ===>
      *************************  TOP OF QUEUE  *******************************
00001 IESUXPL .....              144420000015100-400IEDSIEXS
00002 IESUXPL .....        .........................IEDSIEXS
00003 14:43:54....130 F1..POWSTART.......*....130 F3..VTAMSTRT.......*....130 F
00004 14:43:54....170 F3..VTAMSTRT............170 F2..CICSICCF...2........170 F
00005 14:43:54....220 F3..VTAMSTRT...........!220 F2..CICSICCF...........!220 F
00006 14:43:54....387 F3..VTAMSTRT.......U...U388 F3..VTAMSTRT.......U...U389 F
00007 IESUXPL .....        .........................IEDSIEXS
00008 14:44:09....130 F1..POWSTART.......*....130 F3..VTAMSTRT.......*....130 F
00009 14:44:09....170 F3..VTAMSTRT............170 F2..CICSICCF...2........170 F
00010 14:44:09....220 F3..VTAMSTRT...........!220 F2..CICSICCF...........!220 F
00011 14:44:09....387 F3..VTAMSTRT.......U...U388 F3..VTAMSTRT.......U...U389 F
00012 IESUXPL .....        .........................IEDSIEXS
      *************************  BOTTOM OF QUEUE  *****************************




PF1 : HELP                PF2 : SWITCH HEX/CHAR     PF3 : TERMINATE BROWSE
PF4 : VIEW TOP            PF5 : VIEW BOTTOM         PF6 : REPEAT LAST FIND
PF7 : SCROLL BACK HALF    PF8 : SCROLL FORWARD HALF PF9 : VIEW RIGHT
PF10: SCROLL BACK FULL    PF11: SCROLL FORWARD FULL PF12: UNDEFINED
```

Figure 105. IESCHOUT Display in Character Representation

Figure 106 on page 364 shows a sample CEBR IESCHOUT display in hexadecimal representation:

```
 CEBR           TS QUEUE  IESCHOUT RECORD     1 OF    12    COL     1 OF   348
 ENTER COMMAND ===>
      **************************  TOP OF QUEUE  *******************************
00001 C9C5E2E4E7D7D34000360100004040404040404040404040404040F1F4F4F4F2F0F0F0F0
00002 C9C5E2E4E7D7D34000362100004040404040404040000000000000000000000000000000
00003 F1F47AF4F37AF5F400000000F1F3F040C6F10000D7D6E6E2E3C1D9E3000001350000035C
00004 F1F47AF4F37AF5F400000000F1F7F040C6F30000E5E3C1D4E2E3D9E3000003FA00000868
00005 F1F47AF4F37AF5F400000000F2F2F040C6F30000E5E3C1D4E2E3D9E30000000B00003823
00006 F1F47AF4F37AF5F400000000F3F8F740C6F30000E5E3C1D4E2E3D9E300000003000014E4
00007 C9C5E2E4E7D7D34000362100004040404040404040000000000000000000000000000000
00008 F1F47AF4F47AF0F90000000FF1F3F040C6F10000D7D6E6E2E3C1D9E3000001350000035C
00009 F1F47AF4F47AF0F90000000FF1F7F040C6F30000E5E3C1D4E2E3D9E3000003FA00000868
00010 F1F47AF4F47AF0F90000000FF2F2F040C6F30000E5E3C1D4E2E3D9E30000000B00003823
00011 F1F47AF4F47AF0F90000000FF3F8F740C6F30000E5E3C1D4E2E3D9E300000003000014E4
00012 C9C5E2E4E7D7D34000360200004040404040404040000000000000000000000000000000
      *************************  BOTTOM OF QUEUE  *****************************




PF1 : HELP               PF2 : SWITCH HEX/CHAR     PF3 : TERMINATE BROWSE
PF4 : VIEW TOP           PF5 : VIEW BOTTOM         PF6 : REPEAT LAST FIND
PF7 : SCROLL BACK HALF   PF8 : SCROLL FORWARD HALF PF9 : VIEW RIGHT
PF10: SCROLL BACK FULL   PF11: SCROLL FORWARD FULL PF12: UNDEFINED
```

Figure 106. IESCHOUT Display in Hexadecimal Representation

Flow of Events

The flow of events for a system activity measurement cycle, for example, is as follows:

1. A first time invocation of IESDALOG (start of measurement cycle) is initiated by transaction IEXM and indicated by X’01’ in field UXPLEC of the CICS TS COMMAREA. IESDALOG responds by writing message ″IESDALOG first″ to the console and by passing return code X’00’ to IEXM: continue measurement cycle.
2. Transaction IEXM calls transaction IEXA for measuring the system activity.

   After the measurement, IEXA calls exit program IESDALOG. X'11' in field UXPLEC in the CICS TS COMMAREA indicates that the call is from IEXA.

   IESDALOG saves the current measurement data (stored in IESAIEXA) in temporary storage queue IESDAOUT. Exit program IESDALOG writes message ″IESDALOG measure″ to the console and passes return code X'00' to IEXA: continue taking measurements.
3. A last time invocation of IESDALOG (end of measurement cycle) is indicated by X’02’ in field UXPLEC of the CICS TS COMMAREA. IESDALOG responds by writing message ″IESDALOG last″ to the console.

Error Processing

Errors recognized by the starting transaction IEXM are, for example, syntax errors of the input parameters and CICS TS GETMAIN problems. Error information is passed to one or both user exit programs (IESDALOG and IESCHLOG) depending on the measurements currently being taken.

Error codes (in decimal) are passed in field UXPLERRC of the CICS TS COMMAREA to the user exit program. If necessary, additional information is contained in UXPLERRI. In the following error code list, the kind of additional information is appended in parentheses.

If a link to a user exit program is not possible, the task abends with the CICS TS abend code IESX.

Error Code	Explanation
0	Function completed successfully
1	Unexpected CICS TS error (EIBFN, EIBRCODE)
4	Input data is longer than 100 bytes
5	Input data not found
8	Syntax error (invalid string)
9	Invalid keyword (invalid keyword)
10	Invalid DEVRANGE specification (invalid DEVRANGE)
11	Invalid time specification (invalid time)
12	Duplicate parameter (parameter)
13	Invalid EXIT specification
16	The range limits must be from low to high (invalid DEVRANGE)
17	STARTTIME must be lower than STOPTIME
18	Interval smaller than 10 seconds (invalid INTERVAL)
20	Measure system activity attempted to start twice
21	Measure channel/device activity attempted to start twice
22	Measure system activity canceled
23	Measure channel/device activity canceled
24	Job accounting is not active
25	Internal error, unable to continue, dump taken (dump name)
48	Measure channel/device activity: no SIOs found for range
49	Measure channel/device activity: given range too large, device list incomplete

Figure 107. Error Codes Passed to the User Exit Program

Format of Measurement Data

The following sections show the layout and the format of the logging records as they are stored in the CICS TS temporary storage queues.

Format of System Activity Data

At least one record is available in the CICS TS temporary storage queue IESAIEXA for each single measurement sample. This record contains the measurement data for the **whole system** and the data for **static partitions**.

If **dynamic classes** are defined for a z/VSE system, another record follows containing measurement data for dynamic classes. If **dynamic partitions** are active, more records are created. They provide performance details about dynamic partitions (one record for each dynamic class with active dynamic partitions).

Format of Static Partitions Data

This record format provides:
- Data about overall system performance.
- Detailed performance data about static partitions.
- A summary of data about dynamic classes.

The record format reserves space for 12 static partitions and 23 dynamic classes. If there are fewer static partitions or dynamic classes defined, the rest of the record is filled with binary zeroes and blanks.

Notes:

1. In the following figures, "d" in the Display column indicates that this information is also provided by the activity dialogs described in the *z/VSE Operation* manual.
2. In the following figures, the first byte of LDPARTID is the reply indicator. It is usually blank, but contains an asterisk, if there is an open reply at the console for the corresponding partition.

```
Name     Type     Display  Offset  Explanation
________________________________________________________________________________

*        DATE/TIME INFORMATION
*
LOGREC   DS    0D                  BEGINNING OF THE LOG RECORD
LQID     DS    CL8         000     LOG RECORD QID            (EBCDIC)
LDATE    DS    CL8         008     DATE IN FORMAT:  MM/DD/YY  (EBCDIC)
LTIME    DS    CL8    d    010     DISPLAY TIME:    HH:MM:SS  (EBCDIC)
LTIME2   DS    XL8         018     BEGIN THIS INTERVAL    (STCK BINARY)
LSECS    DS    XL4    d    020     ACTUAL INTERVAL IN SECS     (BINARY)
         SPACE 2
*
*        CPU UTILIZATION
*
LCPUNUM  DS    XL2    d    024     NUMBER OF CPUS IN CEC       (BINARY)
LCPUACT  DS    XL2    d    026     NUMBER OF ACTIVE CPUS       (BINARY)
LCPUQUI  DS    XL2    d    028     NUMBER OF QUIESCED CPUS     (BINARY)
         DS    0F                  FORCE ALIGNMENT
LCPU     DS    XL2    d    02C     SYSTEM CPU TIME IN %       (BINARY)
LWAIT    DS    XL2         02E     SYSTEM WAIT TIME IN %      (BINARY)
LIORATE  DS    XL4    d    030     SYSTEM SIO RATE             (BINARY)
         SPACE 2
*
*        SYSTEM PAGING ACTIVITY
*
LPAGEOUT DS    XL4    d    034     SYSTEM PAGE OUTS            (BINARY)
LOUTRATE DS    XL4    d    038     PAGE-OUT RATE PER SEC       (BINARY)
LPAGEIN  DS    XL4    d    03C     SYSTEM PAGE INS             (BINARY)
LINRATE  DS    XL4    d    040     PAGE-IN RATE PER SEC        (BINARY)
         SPACE 2
*
*        CICS TASK/STORAGE CONTROL DATA
*
LMAXTASK DS    CL4         044     CURRENT MAXTASKS (MXT) LIMIT
*                                  *                 (XMGMXT)  (BINARY)
LPEAKACT DS    CL4    d    048     PEAK # ACTIVE USER TRANSACTIONS
*                                                    (XMGPAT)  (BINARY)
```

Figure 108. Format of Static Partitions Data (Part 1 of 2)

```
LMXTLIMI DS    CL4    d    04C     TIMES MXT LIMIT REACHED
*                                  *                 (XMGTAMXT)(BINARY)
LMXTLIM2 DS    CL4         050     # USER TRANSACTIONS AT MXT LIMIT
*                                                    (XMGTDT)  (BINARY)
LTASKCNT DS    CL2         054     CURR. TASK COUNT  (DSGCNT  )
*                                  *                           (BINARY)
LTASKMAX DS    CL2         056     MAX. TASK ACCUM.  (DSGPNT  )
*                                  *                           (BINARY)
LTASKNUM DS    CL4    d    058     NO. ACTIVE USER TRANSACTIONS
*                                  *                 (XMGTAT)  (BINARY)
LTASKRAT DS    CL4    d    05C     CICS TASKS PER SEC(TENTHS)  (BINARY)
LDISPTCH DS    CL4    d    060     CICS TASKS DISPATCHABLE     (BINARY)
LSUSPEND DS    CL4    d    064     CICS TASKS SUSPENDED        (BINARY)
          SPACE 2
*
*   CICS VSAM FCT STATISTICS (removed during CICS/ESA adaptions)
*
         DS    CL10        068     unused
         SPACE 2
*
*        VSE PARTITION/DYN CLASS DATA PER JOB ACCOUNTING TABLE
 *
LNPART   DS    XL2         072     SAVE NUMBER OF PARTITIONS + CLASSES
         SPACE 1
*        LOCAL STATISTICS TABLE FOR 12 PARTITIONS AND 23 CLASSES
         DS    0F          074     FORCE WORD ALIGNMENT
LPARTAB  EQU    *                  PARTITION STATISTICS TABLE
         DS    CL2         074     dummy for alignment
LDPARTID DS    CL3    d    076     PARTITION/CLASS ID + REPLY IND
LSPACEID DS    CL1    d    079     ADDRESS SPACE ID
LPIBFLAG DS    CL2         07A     PIB FLAG
LJOBNAME DS    CL8    d    07C     JOB NAME
LPHANAME DS    CL8    d    084     PHASE NAME
LPERCENT DS    XL4    d    08C     CPU UTILIZATION IN %
LSIORATE DS    XL4         090     SIO RATE/SECOND
LCPUOVHT DS    XL4    d    094     CPU OVERHEAD TIME
LCPUTIME DS    XL4    d    098     CPU TIME IN 1/300 SECONDS
LCPUFLAG DS    CL1         09C     unused
LIOCOUNT DS    XL4    d    09D     I/O COUNT
LIOCFLAG DS    CL1         0A1     unused
LELAPSET DS    CL6         0A2     JOB START TIME FOR ELAPSE CALC
LELAPTIM DS    XL4    d    0A8     JOB ELAPSED TIME IN SECONDS
         DS    0F          0AC     FORCE WORD ALIGNMENT
LTABSIZE EQU   *-LPARTAB    38     STATISTICS TABLE SIZE
         DS    CL(34*LTABSIZE)     BUILD DISPLAY STATISTICS
LOGEND   EQU   *-LOGREC    544
*        END OF LOG RECORD
```

Figure 108. Format of Static Partitions Data (Part 2 of 2)

Format of Dynamic Classes/Dynamic Partitions Data

There are two record layouts, one is used for data on dynamic classes, the other for data on dynamic partitions of a specific dynamic class.

The following figure shows the record layout for data on dynamic classes.

```
Name     Type     Display  Offset  Explanation
_____________________________________________________________________________

*        LOCAL STATISTICS TABLE FOR 35 ENTRIES (PARTITIONS/CLASSES),
*        SERVES AS REPOSITORY FOR JOB ACCOUNTING DATA/DISPLAY DATA
_____________________________________________________________________________

LNDATA   DS    XL2         000     NUMBER OF ENTRIES
*
*        VSE PARTITION DATA PER JOB ACCOUNTING TABLE
         DS    0F          004     FORCE WORD ALIGNMENT
LPARTAB  EQU    *                  PARTITION STATISTICS TABLE
         DS    CL3         004     dummy for alignment
LDPARTID DS    CL3         007     PARTITION ID  here: CLASS ID
LPIBFLAG DS    CL2         00A     unused
LJOBNAME DS    CL8         00C     unused
LPHANAME DS    CL8         014     unused
LPERCENT DS    XL4    d    01C     CPU UTILIZATION IN %
LSIORATE DS    XL4         020     SIO rate/second
LCPUOVHT DS    XL4    d    024     CPU OVERHEAD TIME
LCPUTIME DS    XL4    d    028     CPU TIME IN 1/300 SECONDS
LCPUFLAG DS    CL1         02C     unused
LIOCOUNT DS    XL4    d    02D     I/O COUNT
LIOCFLAG DS    CL1         031     unused
LELAPSET DS    CL6         032     unused
LELAPTIM DS    XL4         038     unused
*
LCLASS   DS    CL1    d    03C     DYNAMIC CLASS ID
LCLSFLG  DS    CL1         03D     DYNAMIC CLASS FLAG
LACTIV   DS    XL2    d    03E     ACTIVE PARTITIONS/CLASS
LMAXDP   DS    XL2    d    040     MAX ALLOCATED PARTITIONS/CLASS
         DS    0F          044     FORCE WORD ALIGNMENT
LTABSIZE EQU   *-LPARTAB   040     STATISTICS TABLE SIZE
         DS    CL(34*LTABSIZE)     BUILD DISPLAY STATISTICS
```

Figure 109. Format of Dynamic Classes Data

The following figure shows the record layout for data on all dynamic partitions of one dynamic class.

```
Name     Type     Display  Offset  Explanation
______________________________________________________________________________

*        LOCAL STATISTICS TABLE FOR 35 ENTRIES (PARTITIONS/CLASSES),
*        SERVES AS REPOSITORY FOR JOB ACCOUNTING DATA/DISPLAY DATA
______________________________________________________________________________

LNDATA   DS    XL2         000     NUMBER OF ENTRIES
*
*        VSE PARTITION DATA PER JOB ACCOUNTING TABLE
         DS    0F          004     FORCE WORD ALIGNMENT
LPARTAB  EQU    *                  PARTITION STATISTICS TABLE
         DS    CL3         004     dummy for alignment
LDPARTID DS    CL3    d    007     PARTITION ID
LPIBFLAG DS    CL2         00A     PIB FLAG
LJOBNAME DS    CL8    d    00C     JOB NAME
LPHANAME DS    CL8    d    014     PHASE NAME
LPERCENT DS    XL4    d    01C     CPU UTILIZATION IN %
LSIORATE DS    XL4         020     SIO rate/second
LCPUOVHT DS    XL4         024     CPU OVERHEAD TIME
LCPUTIME DS    XL4         028     CPU TIME IN 1/300 SECONDS
LCPUFLAG DS    CL1         02C     unused
LIOCOUNT DS    XL4    d    02D     I/O COUNT
LIOCFLAG DS    CL1         031     unused
LELAPSET DS    CL6         032     JOB START TIME FOR ELAPSE CALC
LELAPTIM DS    XL4         038     JOB ELAPSED TIME IN SECONDS
*
LCLASS   DS    CL1         03C     unused
LCLSFLG  DS    CL1         03D     unused
LACTIV   DS    XL2         03E     unused
LMAXDP   DS    XL2         040     unused
         DS    0F          044     FORCE WORD ALIGNMENT
LTABSIZE EQU   *-LPARTAB   040     STATISTICS TABLE SIZE
         DS    CL(34*LTABSIZE)     BUILD DISPLAY STATISTICS
                           804
```

Figure 110. Format of Dynamic Partitions Data

Format of Channel and Device Activity Data

CICS TS QUEUE IEDSIEXS contains for every measurement point as many records as necessary for the given device range. Device activity information is given for every device in the device range and for every job using this device. 12 device-job entries fit into 1 TSQUEUE record.

The following figure shows record layout for data on channel and device activity.

```
Name     Type     Display  Offset  Explanation
________________________________________________________________________________

TSREC    DS    0F          000     START OF TSQUEUE RECORD
TSDTTIME DS    CL8         000     MEASUREMENT TIME
TSDFTIME DS    F           008     MEASUREMENT INTERVAL
TSLINE   DS    0F          00C
TSDCUUU  DS    CL3         00C     DEVICE NUMBER
TSDSPOOL DS    CL1         00F     DEVICE SPOOLED
TSDPART  DS    CL2         010     PARTITION ID
*        DS    CL2         012     FILLER
TSDJOBN  DS    CL8         014     JOBNAME
TSDSCUU  DS    F           01C     SIO PER DEVICE
TSDSCUX  DS    F           020     SIO PER CONTROL UNIT
TSDSCXX  DS    F           024     SIO PER CHANNEL
         DS    0F          028     FORCE WORD ALIGNMENT
TSLISIZE EQU   *-TSLINE            TSLINE SIZE
         DS    CL(11*TSLISIZE)     STORAGE FOR NEXT 11 ENTRIES
                           15C
```

Figure 111. Format of Channel/Device Data

Examples of Measurement Data

Example 1: Data of Two Static Partitions and One Dynamic Class

The following figure shows overall system activity data and specific data for two static partitions and one dynamic class.

```
Name     Contents          Contents   Explanation
         (hexadec)         (char/dec)
________________________________________________________________________________

LQID     C9C5E2C1C9C5E7C1  IESAIEXA  Name of the logging CICS TS queue
LDATE    F0F261F1F061F9F5  02/10/00  Date in format  MM/DD/YY
LTIME    F0F87AF5F17AF1F5  08:51:15  Begin of the next interval
                                        (readable form)
LTIME2   AABCDBF43D6C7102            Begin of the next interval
                                        (internal STCK format)
LSECS    00000000F               15  Length of interval between
                                        2 measurement points in seconds
*
```

Figure 112. Example 1: Data of Two Static Partitions and One Dynamic Class (Part 1 of 4)

```
*        CPU UTILIZATION
*
LCPUNUM  0006                       6  Number of CPUs in CEC
LCPUACT  0006                       6  Number of active CPUs
LCPUQUI  0000                       0  Number of quiesced CPUs
         0000                          ... force alignment ...
LCPU     000D                      13  CPU utilization in % (CPU time):
                                         sum of all LPERCENT values of
                                         all partitions and classes
LWAIT    024B                     587  CPU utilization in % (WAIT time):
                                         100 × LCPUACT - LCPU
LIORATE  0000003A                  58  SIO rate:
                                        (SUM.current-SUM.previous) / LSECS
                                          where SUM =
                                            sum of all LIOCOUNT values
                                            of all paritions and classes
*
*        SYSTEM PAGING ACTIVITY
*
LPAGEOUT 00000003                   3  Number of pages put on secondary
                                          storage
LOUTRATE 00000000                   0  LPAGEOUT.current - LPAGEOUT.previous
                                          devided by LSECS
LPAGEIN  00000001                   1  Number of pages put into main
                                          storage
LINRATE  00000000                   0  (LPAGEIN.current - LAPGEIN.previous)
                                          devided by LSECS
*
*        CICS TASK/STORAGE CONTROL DATA
*
LMAXTASK 00000014                20  CURRENT MAXTASKS (MXT) LIMIT
*                                    *                 (XMGMXT)  (BINARY)
LPEAKACT 00000006                 6  PEAK # ACTIVE USER TRANSACTIONS
*                                                      (XMGPAT)  (BINARY)
LMXTLIMI 00000000                 0  TIMES MXT LIMIT REACHED
*                                    *                 (XMGTAMXT)(BINARY)
LMXTLIM2 00000000                 0  # USER TRANSACTIONS AT MXT LIMIT
*                                                      (XMGTDT)  (BINARY)
LTASKCNT 000E                    14  CURR. TASK COUNT  (DSGCNT  )
*                                    *                           (BINARY)
LTASKMAX 000E                    14  MAX. TASK ACCUM.  (DSGPNT  )
*                                    *                           (BINARY)
LTASKNUM 0000003A                58  NO. ACTIVE USER TRANSACTIONS
*                                    *                 (XMGTAT)  (BINARY)
LTASKRAT 0000000A                10  CICS TASKS PER SEC(TENTHS)  (BINARY)
LDISPTCH 00000002                 2  CICS TASKS DISPATCHABLE     (BINARY)
LSUSPEND 00000003                 3  CICS TASKS SUSPENDED        (BINARY)
         SPACE 2
 *
```

Figure 112. Example 1: Data of Two Static Partitions and One Dynamic Class (Part 2 of 4)

```
*
*   CICS VSAM FCT STATISTICS (removed during CICS/ESA adaptions)
*
         00000000000000000000      unused
         SPACE 2
 *        VSE PARTITION/DYN CLASS DATA PER JOB ACCOUNTING TABLE
*
LNPART   0010                   16  Number of entries in job accounting
                                       table
LPARTAB  EQU    *                   Partition statistics table
```

First Example of Static Partition Statistics:

```
         0000                       ... dummy for alignment ...
LDPARTID 40C6F4                 F4 First byte: blank or asterisk
                                    Second and third byte: partition id
                                       or class id + 'FF'
LSPACEID F4                      4  Address space ID
LPIBFLAG F9F0                   90  Partition's PIB flag in characters
                                       82:  stopped
                                       80:  unbatched
                                       00:  active
LJOBNAME D5D640D5C1D4C540   NO NAME  Job name
LPHANAME 4040404040404040           Phase name
LPERCENT 00000000                0  CPU utilization in %:
                                    (CPUDIFF × 100) / (LSECS × 300)
                                       where CPUDIFF =
                                          (LCPUTIME+LCPUOVHT).current -
                                            (LCPUTIME+LCPUOVHT).previous
LSIORATE 00000000                0  SIO rate:
                                    LIOCOUNT.current-LIOCOUNT.previous
                                       divided by LSECS
LCPUOVHT 00000014               20  CPU overhead time in 1/300 seconds
LCPUTIME 0000002B               43  CPU time in 1/300 seconds
LCPUFLAG 40                         unused
LIOCOUNT 00000058               88  Number of start I/Os
LIOCFLAG 40                         unused
LELAPSET 000000000000               Job start time
LELAPTIM 00000000                0  Job elapsed time in seconds
```

Second Example of Static Partition Statistics:

```
         0000                       ... dummy for alignment ...
LDPARTID 40C2C7                 BG First byte: blank or asterisk
                                    Second and third byte: partition id
                                       or class id + 'FF'
LSPACEID F0                      0  Address space ID
LPIBFLAG F0F0                   00  Partition's PIB flag in characters
                                       82:  stopped
                                       80:  unbatched
                                       00:  active
```

Figure 112. Example 1: Data of Two Static Partitions and One Dynamic Class (Part 3 of 4)

```
LJOBNAME C9C5E2E7C4C1C340  IESXDAC   Job name
LPHANAME C1E2D4C1F9F04040  ASMA90    Phase name
LPERCENT 00000009                 9  CPU utilization in %:
                                     (CPUDIFF×100) / (LSECS×*300)
                                        where CPUDIFF =
                                           (LCPUTIME+LCPUOVHT).current -
                                            (LCPUTIME+LCPUOVHT).previous
LSIORATE 0000001F                31  SIO rate:
                                     LIOCOUNT.current-LIOCOUNT.previous
                                        devided by LSECS
LCPUOVHT 00000116               278  CPU overhead time in 1/300 seconds
LCPUTIME 000003E5               997  CPU time in 1/300 seconds
LCPUFLAG 40                          unused
LIOCOUNT 00000497              1175  Number of start I/Os
LIOCFLAG 40                          unused
LELAPSET AA9C09C400C0                Job start time
LELAPTIM 00000025                37  Job elapsed time in seconds

Example of Dynamic Class Statistics:
         0000                        ... dummy for alignment ....
LDPARTID 40E8FF                  Y   First byte: blank or asterisk
                                     Second and third byte: partition id
                                        or class id + 'FF'
LSPACEID 00                          Address space ID
LPIBFLAG F0F0                    00  Partition's PIB flag in characters
                                        82:  stopped
                                        80:  unbatched
                                        00:  active
LJOBNAME 0000000000000000            Job name
LPHANAME 0000000000000000            Phase name
LPERCENT 00000003                 3  CPU utilization in %:
                                     (CPUDIFF×100) / (LSECS×300)
                                        where CPUDIFF =
                                          (LCPUTIME+LCPUOVHT).current -
                                           (LCPUTIME+LCPUOVHT).previous
LSIORATE 00000019                25  SIO rate:
                                     LIOCOUNT.current-LIOCOUNT.previous
                                        devided by LSECS
LCPUOVHT 00000078               120  CPU overhead time in 1/300 seconds
LCPUTIME 000000BC               188  CPU time in 1/300 seconds
LCPUFLAG 40                          unused
LIOCOUNT 0000036B               875  Number of start I/Os
LIOCFLAG 40                          unused
LELAPSET 000000000000                Job start time
LELAPTIM 00000000                    Job elapsed time in seconds
```

Figure 112. Example 1: Data of Two Static Partitions and One Dynamic Class (Part 4 of 4)

Example 2: Data of One Dynamic Class/One Dynamic Partition

The following figure shows data of one dynamic class.

```
Name     Content          Content    Explanation
         (hexadec)        (char/dec)
______________________________________________________________________________

LNDATA   0004                   4   Number of entries
*
*        VSE PARTITION DATA PER JOB ACCOUNTING TABLE
         0000                       ... dummy for alignment ...
LDPARTID 40E8FF                 Y   First byte: blank
                                    Second and third byte:
                                        class id + 'FF'
LPIBFLAG 0000                       unused
LJOBNAME 0000000000000000           unused
LPHANAME 0000000000000000           unused
LPERCENT 00000003                3  CPU utilization in %:
                                    (CPUDIFF×100) / (LSECS×300)
                                       where CPUDIFF =
                                         (LCPUTIME+LCPUOVHT).current -
                                          (LCPUTIME+LCPUOVHT).previous
LSIORATE 00000019               25  SIO rate:
                                    LIOCOUNT.current-LIOCOUNT.previous
                                       devided by LSECS
LCPUOVHT 00000078              120  CPU overhead time in 1/300 seconds
LCPUTIME 000000BC              188  CPU time in 1/300 seconds
LCPUFLAG 40                         unused
LIOCOUNT 0000036B              875  Number of start I/Os
LIOCFLAG 40                         unused
LELAPSET 000000000000               unused
LELAPTIM 00000000                   unused
*
LCLASS   E8                      Y  Dynamic class id
LCLSFLG  80                         Class table entry flag:
                                    x'80' - dynamic class enabled
                                    x'40' - class table entry in error
LACTIV   0001                    1  Active partitions/class
LMAXDP   0008                    8  Max. allocated partitions/class
```

Figure 113. Example 2: Data of One Dynamic Class

The following figure shows data of a single dynamic partition.

```
Name     Content          Content    Explanation
         (hexadec)        (char/dec)
_________________________________________________________________________________

*        LOCAL STATISTICS TABLE FOR 35 ENTRIES (PARTITIONS/CLASSES),
*        SERVES AS REPOSITORY FOR JOB ACCOUNTING DATA/DISPLAY DATA
LNDATA   0001                      1  Number of active partitions/class
*        VSE PARTITION DATA PER JOB ACCOUNTING TABLE
         0000                         ... dummy for alignment ...
LDPARTID 40E8F2                   Y2  First byte: blank or asterisk
                                      Second and third byte: partition id
LPIBFLAG F0F0                     00  PIB flag
                                         82:  stopped
                                         80:  unbatched
                                         00:  active
LJOBNAME C9C5E2E7C4C4C340  IESXDDC    Job name
LPHANAME C1E2D4C1F9F04040  ASMA90     Phase name
LPERCENT 00000003                  3  CPU utilization in %:
                                        (CPUDIFF×100) / (LSECS×300)
                                          where CPUDIFF =
                                            (LCPUTIME+LCPUOVHT).current -
                                             (LCPUTIME+LCPUOVHT).previous
LSIORATE 00000019                 25  SIO rate:
                                      LIOCOUNT.current-LIOCOUNT.previous
                                         devided by LSECS
LCPUOVHT 00000078                120  CPU overhead time in 1/300 seconds
LCPUTIME 000000BC                188  CPU time in 1/300 seconds
LCPUFLAG 40                           unused
LIOCOUNT 0000036B                875  Number of start I/Os
LIOCFLAG 40                           unused
LELAPSET AAA079330140                 Job start time
LELAPTIM 00000020                 32  Job elapsed time in seconds
*
LCLASS   00                           unused
LCLSFLG  00                           unused
LACTIV   0000                         unused
LMAXDP   0000                         unused
```

Figure 114. Example 2: Data of One Dynamic Partition

Example 3: Channel and Device Activity Data

The following figure shows channel and device activity data of one device and one job using this device.

```
Name     Content          Content    Explanation
         (hexadec)        (char/dec)
________________________________________________________________________________

TSDTTIME F0F67AF5F67AF5F4  06:56:54  Measurement time
TSDFTIME 0000000F                15  Measurement interval
TSLINE
TSDCUUU  F2F3F0                 230  Device number (cuu)
TSDSPOOL 40                          Device spooled
                                        x'40' - no
                                        x'5C' - yes
TSDPART  E8F2                    Y2  Partition ID
         0000                        ... filler ...
TSDJOBN  C9C5E2D7C4C4C340  IESXDDC   Job name
TSDSCUU  00000151               337  SIO per device
TSDSCUX  00005131             20785  SIO per control unit
TSDSCXX  00005131             20785  SIO per channel
```

Figure 115. Example 3: Channel and Device Activity Data

Appendix A. Fast Paths and Synonyms for Dialogs

This appendix lists the fast paths and synonyms that can be used to access dialogs of the Interactive Interface. For information about creating your own synonyms, refer to "Maintaining Synonyms" on page 125.

In Table 9:

- **z/VSE Selection** is the name used for a dialog in a selection panel of the Interactive Interface (*Analyze and Apply PTFs*, for example).
- **Default For** shows where the dialog appears in the default panel hierarchies supplied by z/VSE:
  - **A** = hierarchy for **SYSA** (type 1 user)
  - **O** = hierarchy for **OPER** (type 2 user)
  - **P** = hierarchy for **PROG** (type 2 user)
  - **S** = hierarchy for **$SRV** (type 2 user)

  A, O, and P identify the default panel hierarchies available for a system administrator (SYSA), an operator (OPER), and a programmer (PROG).

  S, which stands for service ($SRV), identifies a default panel hierarchy which provides access to a selected set of standard dialogs for problem determination. This panel hierarchy is mainly intended for IBM personnel doing remote problem determination for a user site via a data link connecting the user installation with an IBM Support Center, for example. But the $SRV panel hierarchy can also be used for local problem determination.
- **Fast Path** shows the number string that can be used to access the dialog from a user's *initial* selection panel.
- **Default Synonym** shows the character string (if any) that is predefined for the dialog. Users can enter the synonym from *any* selection panel of the Interactive Interface to access the dialog.

  Note that predefined synonyms for groups of dialogs (such as the dialogs for backing up data) are also listed in the figure.

Table 9. Fast Paths and Synonyms for Dialogs

z/VSE Selection	Default For	Fast Path	Default Synonym
Analyze and Apply PTFs	A	1422	---
Apply PTFs	A	1423	---
Archive All ICCF Libraries on Tape	A P O	37242 5652 5242	--- --- ---
Backup	A P O	37 56 5	BACKUP BACKUP BACKUP
Backup a File	A P O	3752 5672 552	--- --- ---
Backup a User Catalog to Disk	A O	37184 5184	--- ---
Backup a User Catalog to Tape	A O	37183 5183	--- ---

Table 9. Fast Paths and Synonyms for Dialogs (continued)

z/VSE Selection	Default For	Fast Path	Default Synonym
Backup a Volume	A P O	3751 5671 551	--- --- ---
Backup History File	A O	373 53	--- ---
Backup the DTSFILE (All ICCF Libraries)	A P O	37241 5651 5241	--- --- ---
Backup the Master Catalog to Disk	A O	37182 5182	--- ---
Backup the Master Catalog to Tape	A O	37181 5181	--- ---
Backup VSAM File	A P O	3713 563 513	--- --- ---
Backup VSE Library on Tape	A O	3721 521	--- ---
Catalog Printer UCB	A	245	UCB
Change Nicknames	A	146	---
Configure Hardware	A	241	---
Console	A P O S	31 51 2 2	CONSOLE CONSOLE CONSOLE ---
Copy a File	A P O	3772 5692 572	--- --- ---
Copy a Volume	A P O	3771 5691 571	--- --- ---
Create Application Job Stream	A P	52 3	--- ---
Create Network Tape	A	242	---
Create Stand-Alone Dump Program on Tape	A P S	461 461 161	--- --- ---
Create Stand-Alone Dump Program on Disk	A P S	462 462 162	--- --- ---
Customize z/VSE Workstation Platform	A	216	---
Define a Library	A P	223 63	--- ---
Define a New File	A P	222 62	--- ---
Define a New User Catalog	A	226	---
Define an Alternate Index or Name	A P	224 64	--- ---

Table 9. Fast Paths and Synonyms for Dialogs (continued)

z/VSE Selection	Default For	Fast Path	Default Synonym
Define Transaction Security	A	28	--- ---
Display Active Users/Send Message	A P O S	33 52 41 4	USERS or MESSAGE ---
Display Channel and Device Activity	A O	362 72	SIO SIO
Display CICS TS Storage	A O	364 74	--- ---
Display or Process a Catalog, Space	A	225	---
Display or Process a File	A P	221 61	--- ---
Display Storage Layout	A O	363 73	--- ---
Display System Activity	A P O	361 55 71	DA DA DA
Display VTOC	A	23	VTOC
Down-Level Check	A	1431	---
Dump Program Utilities	A P S	46 46 16	--- --- ---
Enter News	A P O	34 53 42	NEWS NEWS NEWS
Export-Disconnect a User Catalog	A O	3715 515	--- ---
Export ICCF Library Members to Tape	A P O	37243 5653 5243	--- --- ---
Export VSAM File	A P O	3711 561 511	--- --- ---
Fast Service Upgrade	A	143	---
File and Catalog Management Dialogs	A P	22 6	VSAM VSAM
FlashCopy VSAM Catalog/Files	A O	3719 519	--- ---
Format ICCF Dump Data	A P S	467 467 167	--- --- ---
FSU Installation	A	1433	---
FSU Preparation	A	1432	---
IBM Service	A	14	SERVICE
Import-Connect a User Catalog	A O	3716 516	--- ---

Table 9. Fast Paths and Synonyms for Dialogs (continued)

z/VSE Selection	Default For	Fast Path	Default Synonym
Import ICCF Library Member	A P O	37254 5664 5254	--- --- ---
Import VSAM File	A P O	3712 562 512	--- --- ---
Inspect Dump Management Output	A P S	44 44 14	--- --- ---
Inspect Message Log	A P S	42 42 12	LOG LOG ---
Install Generation Feature	A	13	---
Install Network Tape	A	15	---
Install Program(s) from Tape	A	112	---
Install Programs - V1 Format	A	12	---
Install Programs - V2 Format	A	11	---
Invoke CEDA	A	72	RDO
Invoke CEMS	A	73	CEMS
Invoke CEOS	P O	82 62	CEOS CEOS
Invoke CEMT	A P O	71 81 61	MT MT MT
List and Process User Files in Host Transfer File	A	381	---
Look Up PTF/APAR	A P S	1448 458 158	--- --- ---
Maintain Application Profiles	A	213	APM
Maintain Certificate–User ID List	A	29	---
Maintain Dynamic Partitions	A	27	---
Maintain PRIMARY Sublibraries	A	215	---
Maintain Printer FCB	A	244	FCB
Maintain Selection Panels	A	212	SPM
Maintain Synonyms	A P O	214 57 8	SYNONYMS SYNONYMS SYNONYMS
Maintain User Profiles	A	211	UPM
Maintain VTAM Application Names	A	25	---
Maintain VTAM Startup Options	A	26	---
Manage Batch Queues	A P O S	32 2 3 3	POWER POWER POWER ---

Table 9. Fast Paths and Synonyms for Dialogs (continued)

z/VSE Selection	Default For	Fast Path	Default Synonym
Manage List Queue	A P O S	321 21 31 31	LST LST LST ---
Manage Punch Queue	A P O S	323 23 33 33	PUN PUN PUN ---
Manage Reader Queue	A P O S	322 22 32 32	RDR RDR RDR ---
Manage Transmit Queue	A P O S	324 24 34 34	XMT XMT XMT ---
Manage Wait for Run Subqueue	A P O S	325 25 35 35	--- --- --- ---
Manage In-Creation Queue	A P O S	326 26 36 36	--- --- --- ---
Move Files from Host Transfer File to ICCF	A	385	---
Move Files from Host Transfer File to VSAM	A	383	---
Move ICCF Members to Host Transfer File	A	384	---
Move VSAM Files to Host Transfer File	A	382	---
Online Problem Determination	A P S	41 41 11	OLPD OLPD ---
PC File Transfer	A	386	---
Personal Computer Move Utilities	A	38	IWS
Personalize History File	A	145	---
Prepare for Installation	A	111	---
Print IPL Diagnostics	A P S	466 466 166	--- --- ---
Print SDAID Tape	A P S	468 468 168	--- --- ---
Print Service Document	A	1421	---
Program Development Library	A P O	51 1 1	ICCFS ICCFS ICCFS

Table 9. Fast Paths and Synonyms for Dialogs (continued)

z/VSE Selection	Default For	Fast Path	Default Synonym
Program Development Library (Primary Library)	A P O	511 11 11	ICCF ICCF ICCF
Remove PTF Records from History File	A	1424	---
Remove Stand-Alone Dump Program from a SYSRES Disk	A P O	463 463 163	--- --- ---
Restore	A P O	37 56 5	RESTORE RESTORE RESTORE
Restore a File	A P O	3762 5682 562	--- --- ---
Restore a Member of an ICCF Library	A P O	37253 5663 5253	--- --- ---
Restore a User Catalog from Disk	A O	37174 5174	--- ---
Restore a User Catalog from Tape	A O	37173 5173	--- ---
Restore a Volume	A P O	3761 5681 561	--- --- ---
Restore History File	A O	374 54	--- ---
Restore One ICCF Library	A P O	37252 5662 5252	--- --- ---
Restore the DTSFILE (All ICCF Libraries)	A P O	37251 5661 5251	--- --- ---
Restore the Master Catalog from Disk	A O	37172 5172	--- ---
Restore the Master Catalog from Tape	A O	37171 5171	--- ---
Restore VSAM File	A P O	3714 564 514	--- --- ---
Restore VSE Library from Tape	A O	3722 522	--- ---
Retrace History File	A P S	1441 451 151	--- --- ---
Retrace APARs	A P S	1445 455 155	--- --- ---

Table 9. Fast Paths and Synonyms for Dialogs (continued)

z/VSE Selection	Default For	Fast Path	Default Synonym
Retrace Component ID	A P S	1447 457 157	--- --- ---
Retrace Components	A P S	1443 453 153	--- --- ---
Retrace Members	A P S	1446 456 156	--- --- ---
Retrace Products	A P S	1442 452 152	--- --- ---
Retrace PTFs	A P S	1444 454 154	--- --- ---
Retrieve Files from Another System	A P	392 582	--- ---
Retrieve Message	A P O S	35 54 43 5	RETRIEVE RETRIEVE RETRIEVE ---
Scan Dump Files on Tape	A P O	464 464 164	--- --- ---
Scan Dump Files on Disk	A P O	465 465 165	--- --- ---
Scan VSE Library Backup Tape	A O	3723 523	--- ---
Storage Dump Management	A P S	43 43 13	--- --- ---
Submit a Job to Another System	A P	393 583	--- ---
Tailor IPL Procedure	A	243	---
TCP/IP Configuration	A	246	---
Transfer Files and Jobs to Another System	A P	39 58	NET NET
Transfer Files to Another System	A P	391 581	--- ---
Verify Location of Involved Serviced Files	A	141	---
FlashCopy VSAM Catalog / Files	A	3719	---

Appendix B. Format of DTSECTAB Macro for Type User

This macro is mainly intended to change user FORSEC in DTSECTAB. Other user information is not recommended to keep in DTSECTAB. The VSE.CONTROL.FILE is the correct place to keep user information.

```
►►──DTSECTAB TYPE=USER,NAME=username──┬──────────────────────┬────────────►
                                      └─,PASSWRD=password─┘

                                                         ┌─,AUTH=NO──┐
►──┬─────────────────────────────────────────────────┬──┼───────────┼─────►
   │          ┌─,───────────────────────────┐          │  └─,AUTH=YES─┘
   │          │                  ┌─,READ──┐ │          │
   └─,ACC=(───▼─┬─class───────┬──┼────────┼─┴─)───────┘
                └─class-class─┘  └─,right─┘

                     ┌─,READDIR=NO──┐  ┌─,MCONS=NO──┐
►──┬───────────────┬─┼──────────────┼──┼────────────┼──────────────────────►
   └─,RIGHT=BTRANS─┘ └─,READDIR=YES─┘  └─,MCONS=YES─┘

►──┬─────────────────────────┬─────────────────────────────────────────────►◄
   └─,SUBTYPE=──┬─INITIAL─┬──┘
                └─FINAL───┘
```

When two or more entries with identical TYPE and NAME specifications exist, it is unpredictable which profile will be used by Access Control.

Mandatory Parameters:

TYPE
Indicates the kind of profile (access definition) to be made with the DTSECTAB macro call. Specify USER for a user profile definition.

NAME
Indicates for which user this profile is to be defined. For **username**, specify the appropriate user identification of 4 to 8 characters (VSE/ICCF user IDs cannot have more than 4 characters). The user ID associated with a batch job (specified in the * $$ JOB statement, for example, or in the // ID statement) is checked against this 'username.'

Optional Parameters:

PASSWRD
Specifies the 3 to 6-character password associated with the user identification specified in the NAME parameter.

This parameter **must be coded** if you anticipate that security information (user ID and password) will be included in a submitted job.

Normally, there is no need to have this parameter in DTSECTAB when jobs are submitted from *secure environments* such as the z/VSE Interactive Interface, VSE/ICCF or the Workstation File Support. In these cases, password validation is omitted because the jobs are considered *authenticated*. Only if the job does include user ID and password for some special reason (for example, to have it run with a user ID that is different from the submitting user), the password must be available in DTSECTAB.

ACC Specifies the access control classes to which the user is to have access. For **class**, specify a decimal number from 1 to 32, corresponding to the ACC value of the resource(s) to be accessed. You can also specify a list of classes, for example (1,3,5), or a range of classes, for example (1-3). For **right**, specify the access right which the user is to have to this class. Possible values are:

ALT for alter,
UPD for update,
READ for read-only (the default access right), and
CON for connect.

For more information on access rights, see Table 3 on page 152.

Examples:

ACC=(1-3,READ,5,ALT)
would give the user the right to read resources with the classes 1, 2 and 3, and to alter resources with class 5.

ACC=(1-32)
would give the user read-only access to all access controlled resources (READ is the default access right).

This parameter is not necessary if AUTH=YES is specified.

AUTH
Specifying **YES** identifies the user named in the NAME parameter as a security administrator. This user has unrestricted access to all protected resources. If the AUTH parameter is omitted, AUTH=NO is assumed. Note that **several** users can have AUTH=YES.

AUTH=YES includes master console authorization (see parameter MCONS).

RIGHT
Specifying RIGHT=BTRANS gives the user the capability to catalog, rename and delete B-transients in an access-controlled sublibrary. To catalog, rename or delete a member which is a B-transient, a user must have:

- The appropriate access right to the member, **and**
- RIGHT= BTRANS in his user profile.

Note: B-transients can be cataloged only into protected sublibraries and executed only if they are stored in a protected sublibrary. Access control considers any phase beginning with $$B... as a B-transient.

READDIR
Specifying **YES** indicates that a user who has the access right of CON to a library or sublibrary is allowed to read the respective directory information, for example the names of all members (but not the contents).

MCONS
YES indicates that a user has *master console* authorization. MCONS=NO means that a user has *user console* authorization only. AUTH=YES includes master console authorization; in this case, the MCONS specification is meaningless.

The default is AUTH=NO and MCONS=NO which means user console authorization.

When system security (IPL parameter SEC=YES) is active, MCONS=YES in DTSECTAB affects the authorization of the user to enter input via the VM/VSE Interface and to submit CP commands via JCL * CP statements. In these cases, however, the MCONS=YES option in DTSECTAB is only effective in combination with the DTSECTAB option AUTH=NO. The option AUTH=YES, which is set in DTSECTAB for user type 1 (SYSA), provides unrestricted input authorization, regardless of the MCONS option.

SUBTYPE

This parameter is required in the first and the last call of the DTSECTAB macro, and is not allowed in the intermediate calls. The value **INITIAL** must be used in the first call, and the value **FINAL** in the last call.

Appendix C. Batch Program IESUPDCF

The batch utility program **IESUPDCF** allows the system administrator to maintain user profiles in the VSE Control File (IESCNTL) and in the VSE/ICCF DTSFILE. With this program, you can **ADD, ALT**er, and **DEL**ete user profiles. IESUPDCF helps you save time when configuring user profiles.

Preparing to Use IESUPDCF

The following section describes the procedures you should perform before using IESUPDCF.

Planning for User Profiles

With z/VSE you can use three types of user profiles.

A VSE/ICCF (short form: ICCF) user profile is a type 1 or type 2 user profile with a 4 character user ID. It is defined in the VSE control file (IESCNTL) and also in the VSE/ICCF DTSFILE.

Model profiles for type 1 and type 2 user profiles are provided:

Type 1 User Profile
: Valid for the System Administrator. Access to all z/VSE functions, including ICCF.

Type 2 User Profile
: Valid for Operators and Programmers. Access to most of the z/VSE functions, including ICCF.

Type 3 User Profile
: Valid for general users (and Type 1 and Type 2 users with a user ID of 4 to 8 characters). Access to selected functions, but not to ICCF.

Information for ICCF users is recorded in the IESCNTL and in the DTSFILE. Information for type 3 user profiles is only recorded in the IESCNTL file. For the following discussion you should know that ICCF-related definitions (PASSWORD and LIBRARY) are recorded in two places: in the DTSFILE and in the IESCNTL file.

Figure 116 on page 396 shows skeleton IESUPDCF. It is shipped in ICCF library 59. You have to change this skeleton to add, alter, or delete user profiles. Before you change skeleton IESUPDCF, you should carefully plan for the types of users you want to create.

Preparing Skeleton IESUPDCF

You have to prepare skeleton IESUPDCF according to your needs. This may include:

- Set the **ICCF** parameter for all users referred to in the job.
- Insert **ADD** statements for adding user profiles.
- Insert **ALT**er statements for altering user profiles.
- Insert **DEL**ete statements for deleting user profiles.

The following sections have more details.

Setting the ICCF Parameter in Skeleton IESUPDCF

With the setting of the ICCF parameter, you control the generation of job DTRUPD, which updates the DTSFILE. You must enter either Yes, No, or Ignore. There is no default.

```
►►──ICCF=──┬─Yes────┬──────────────────────────────────────────────►◄
           ├─No─────┤
           └─Ignore─┘
```

ICCF=YES
: IESUPDCF updates user profiles in the control file (IESCNTL). For ICCF users, IESUPDCF updates user profiles in the DTSFILE. Therefore, a new job DTRUPD is generated.

  The following describes how specifying ICCF=YES affects the ADD, ALTER, and DELete statements:

  ADD The new user is added to the IESCNTL control file. The definitions of the model user profile are used as default.

  If the model profile is for an ICCF user and the new user ID is 4 characters long, then the new user will also be an ICCF user. Thus, the DTSUTIL statement is generated for job DTRUPD.

  ALTer The user definition is altered in the IESCNTL control file. If the user profile is for an ICCF user, a DTSUTIL statement is generated for job DTRUPD.

  DELete
  The user definition in the IESCNTL control file is deleted. If the user profile is an ICCF user, the DTSUTIL statement for job DTRUPD is generated.

ICCF=No
: No update of the DTSFILE is performed. This means that you cannot ADD or DELete ICCF users. In addition, you cannot ALTer the password or the ICCF library of ICCF users.

  The following describes how specifying ICCF=NO affects the ADD, ALTer, and DELete statements:

  ADD If the model profile is for an ICCF user, then the ADD statement is ignored, and an error message is inserted into the listing. If the model profile is not for an ICCF user, the new user is added to the IESCNTL control file.

  ALTer ICCF-related definitions (PASSWORD and LIBRARY) are not altered in the IESCNTL control file.

DELete
If the user profile is an ICCF user, the statement is ignored and an error message is inserted into the listing. For type 3 user profiles, the definition in the control file is deleted.

ICCF=Ignore
You can ADD, ALTer, or DELete any user in the control file (IESCNTL). For VSE/ICCF users, however, the DTSFILE is not updated. IGNORE must be used if the control file is not related to an ICCF subsystem. With VSE/ESA 2.4, however, a single VSE Control File is recommended.

The following describes how specifying ICCF=IGNORE affects the ADD, ALTer, and DELete statements:

ADD The new user is added to the control file.

ALTer All specified parameters are altered in the control file. If specified, the LIBRARY parameter is ignored, since it is only relevant for ICCF subsystems.

DELete
The user definition in the control file is deleted, independent of the user profile type.

Refer also to item 2 in skeleton IESUPDCF, shown in Figure 116 on page 396.

Adding a User ID in Skeleton IESUPDCF

To **ADD** a user ID, you insert the following statement into skeleton IESUPDCF:

```
►►──ADD user_id,password,profile──┬─────────────────┬──┬────────────┬──────────►
                                   └─,Catalog=catname─┘  └─,Days=nnn─┘

►──┬──────────────────┬──┬─────────────────────────┬──┬──────────────────────┬──►
   └─,Library=libname─┘  └─,Initial=─┬─nnnnnnnn(A)─┤  └─,Operator=oper_id─┘
                                     └─nnnnnnnn(S)─┘

►──┬────────────────────┬──┬───────────────────┬──┬──────────────┬─────────────►
   └─,PRiority=priority─┘  └─,Synonym=user_id─┘  └─,Timeout=mm─┘

►──┬──────────────┬──┬──────────────┬──┬──────────────┬──┬──────────────┬─────►
   └─,APM=─┬─Yes─┤  └─,BAT=─┬─Yes─┤  └─,BQA=─┬─Yes─┤  └─,CMD=─┬─Yes─┤
           └─No──┘          └─No──┘          └─No──┘          └─No──┘

►──┬──────────────┬──┬──────────────┬──┬──────────────┬──┬──────────────┬─────►
   └─,COD=─┬─Yes─┤  └─,COU=─┬─Yes─┤  └─,DVF=─┬─Yes─┤  └─,ESC=─┬─Yes─┤
           └─No──┘          └─No──┘          └─No──┘          └─No──┘

►──┬──────────────┬──┬────────────────┬──┬───────────────┬──┬────────────┬────►
   └─,MVC=─┬─Yes─┤  └─,Natlang=─┬─E─┤  └─,NEWS=─┬─Yes─┤  └─,OID=name─┘
           └─No──┘              ├─C─┤           └─No──┘
                                ├─G─┤
                                └─K─┘

►──┬───────────────┬──┬──────────────┬──┬─────────────────────────┬───────────►
   └─,OLPD=─┬─Yes─┤  └─,PSL=─┬─Yes─┤  └─,Revoke=─┬─mm/dd/yy─┤
            └─No──┘          └─No──┘             └─0────────┘

►──┬──────────────┬──┬──────────────┬──┬──────────────┬───────────────────────►◄
   └─,SPM=─┬─Yes─┤  └─,UPM=─┬─Yes─┤  └─,XRF=─┬─Yes─┤
           └─No──┘          └─No──┘          └─No──┘
```

The first three parameters are mandatory; the rest are optional. **Do not change the order of the mandatory parameters**.

Mandatory Parameters

Note: Each ADD statement may use one or more physical lines. A continuation line is indicated by the continuation character "–" as the last character in the previous line. The continuation character must be preceded by a blank or a comma. The required parameters must be specified together with the ADD statement on *one* line.

user-id
: The user ID which identifies the user to the system. It must be 4-8 alphanumeric characters long and may include the characters @, #, or $. Blanks are not allowed.

  Note: For ICCF users, the user ID can only be 4 characters long.

password
: This is the password associated with the user ID. It can be 3 - 8 alphanumeric characters long and may include the characters @, #, or $. Blanks are not allowed.

profile
This is the identification (user ID) of a user already defined to the system and used as a model for the new user. It must be 4-8 characters long. Using optional parameters you can alter the defaults for the new user ID.

Optional Parameters

This section describes the optional parameters you can use with either the ADD statement, or the ALTer statement (described on page 395).

Catalog=*catname*
The name of the user's default VSE/VSAM catalog (IJSYSCT). This parameter is not available for type 3 users.

Days=*nnn*
The number of days before the user's password expires. Specify a number between 0 and 365. If you enter 0, the password will **not** expire.

Library=*libname*
The user's primary ICCF library. This value may be 4 digits in length. When specifying:
- ICCF=IGNORE, the LIBRARY parameter is ignored, since it is only relevant for ICCF subsystems.
- ICCF=NO, you cannot change the library for an VSE/ICCF user.

Initial=*nnnnnnnn(A)***|***nnnnnnnn(S)*
Initial function performed at sign on. You can use up to eight alphanumeric characters. The value must be followed by the type specification:
- (A) - if the initial function is an application, or
- (S) - if the initial function is a selection panel.

For example:

`INITIAL=FUNCNAME(A)` for an application.

Operator=*oper-id*
3-character operator identification for CICS. The ID must be unique.

PWD|PAssword=*password*
It can be 3 - 8 alphanumeric characters long and may include the characters @, #, or $. Blanks are not allowed. You cannot change the password for an ICCF user when specifying ICCF=NO. This parameter is only to be used together with the ALTer statement.

PRiority=*priority*
The value which CICS uses for the dispatching priorities of the user. Enter a number from **0 - 255**. 0 is the highest priority; 255 the lowest.

Synonym=*user-id*
This defines the user ID to be used as a model for synonyms. z/VSE provides synonyms for users SYSA, PROG, and OPER. These can be used as models for other users.

Timeout=*mm*
Gives the value in minutes used by CICS to initiate sign off after the value specified has elapsed since the latest terminal activity. You can specify a value from 0 to 60. The value you specify is always rounded up to a multiple of 5 minutes. A value of 0 means no time out. 0 should be specified for ICCF users.

APM=Yes|No
When set to “Yes”, you can create and maintain application profiles. This is only valid for the administrator (type 1) user profiles.

BAT=Yes|No
When set to “Yes”, you can submit jobs for batch processing.

BQA=Yes|No
When set to “Yes”, you can manage all VSE/POWER jobs of an ICCF user.

CMD=Yes|No
When set to “Yes”, you can enter system console commands from the *System Console* dialog. This authorization is **not** available for general (type 3) user profiles.

COD=Yes|No
When set to “Yes”, you will be asked for confirmation when you try to delete ICCF members.

COU=Yes|No
When set to “Yes”, all console output is shown.

DVF=Yes|No
When set to “Yes”, you can define and delete VSE/VSAM files, libraries, alternate indexes, and alternate names. This authorization is **not** possible for general (type 3) user profiles.

ESC=Yes|No
When set to “Yes”, you can *escape* to CICS. This lets you leave the Interactive Interface and work directly with CICS.

MVC=Yes|No
When set to “Yes”, you can process VSE/VSAM catalogs and define and delete VSE/VSAM space. This authorization is **not** available for general (type 3) user profiles.

Natlang=E|C|G|K
National language indicator for this user.
E = English, C = Chinese, G = German, K = Kanji

NEWS=Yes|No
When set to “Yes”, the system displays*news items* to you. News items are messages which the system displays when you sign on or when you are already signed on.

OID=*name*
Name of the terminal operator. Can be 1 to 20 characters long and must be unique for each user. Cannot terminate with a blank. Must match exactly the operator name entered in the CSSN sign-on transaction.

OLPD=Yes|No
When set to “Yes”, you can delete Online Problem Determination (OLPD) incident records from the system. This authorization is **not** available for general (type 3) user profiles.

PSL=Yes|No
When set to “Yes”, you will have a private sublibrary (primary user ID).

Revoke=*mm/dd/yy***|0**
Revoke date when the user ID will be revoked. If zero is specified, the user ID will never get revoked.

SPM=Yes|No
: When set to "Yes", you can create and maintain selection panels. This is only valid for the administrator (type 1) user profiles.

UPM=Yes|No
: When set to "Yes", you can create and maintain user profiles. This is only valid for the administrator (type 1) user profiles.

XRF=Yes|No
: When set to "Yes", the user gets signed off after XRF takeover. Otherwise, the user stays signed on.

Altering a User ID in Skeleton IESUPDCF

To **ALT**er a user ID, you insert the following statement into skeleton IESUPDCF:

```
►►──ALTer userid──┬──────────────────┬──┬──────────┬──┬──────────────────┬──────►
                  └─,Catalog=catname─┘  └─,Days=nnn─┘  └─,Library=libname─┘

►──┬─────────────────────────────┬──┬───────────────────┬──┬───────────────────┬──►
   └─,Initial=──┬─nnnnnnnn(A)─┬─┘  └─,Operator=oper-id─┘  ├─,PWD=password─────┤
                └─nnnnnnnn(S)─┘                           └─PAssword=password─┘

►──┬──────────────────────┬──┬──────────────────┬──┬─────────────┬────────────────►
   └─,PRiority=priority───┘  └─,Synonym=user-id─┘  └─,Timeout=mm─┘

►──┬─────────────────┬──┬─────────────────┬──┬─────────────────┬──┬─────────────────┬──►
   └─,APM=──┬─Yes─┬─┘  └─,BAT=──┬─Yes─┬─┘  └─,BQA=──┬─Yes─┬─┘  └─,CMD=──┬─Yes─┬─┘
            └─No──┘             └─No──┘             └─No──┘             └─No──┘

►──┬─────────────────┬──┬─────────────────┬──┬─────────────────┬──┬─────────────────┬──►
   └─,COD=──┬─Yes─┬─┘  └─,COU=──┬─Yes─┬─┘  └─,DVF=──┬─Yes─┬─┘  └─,ESC=──┬─Yes─┬─┘
            └─No──┘             └─No──┘             └─No──┘             └─No──┘

►──┬─────────────────┬──┬──────────────────┬──┬──────────────────┬──┬─────────────┬──►
   └─,MVC=──┬─Yes─┬─┘  └─,Natlang=──┬─E─┬─┘  └─,NEWS=──┬─Yes─┬─┘  └─,OID=name─┘
            └─No──┘                 ├─C─┤              └─No──┘
                                    ├─G─┤
                                    └─K─┘

►──┬──────────────────┬──┬─────────────────┬──┬───────────────────────────┬──────►
   └─,OLPD=──┬─Yes─┬─┘  └─,PSL=──┬─Yes─┬─┘  └─,Revoke=──┬─mm/dd/yy─┬─┘
             └─No──┘             └─No──┘                └─0────────┘

►──┬─────────────────┬──┬─────────────────┬──┬─────────────────┬──────────────►◄
   └─,SPM=──┬─Yes─┬─┘  └─,UPM=──┬─Yes─┬─┘  └─,XRF=──┬─Yes─┬─┘
            └─No──┘             └─No──┘             └─No──┘
```

ALTer checks the user types and performs those changes allowed for the specific user types. For the ALTer statement the first parameter (*userid*) is mandatory. The user ID identifies the user to the system. It must be 4-8 alphanumeric characters long and may include the characters @, #, or $. Blanks are not allowed. You also must specify at least **one** additional parameter from the optional parameters shown in the ALTer statement syntax. The optional parameters you can use are described in detail under "Optional Parameters" on page 393. You may have more than one ALTer statement for the same user profile.

Each ALTer statement may use one or more physical lines. A continuation line is indicated by the continuation character "–" as the last character in the previous line. The continuation character must be preceded by a blank or a comma. The *user-id* must be specified on the same line as the ALTer statement.

Deleting a User ID in Skeleton IESUPDCF

To **DEL**ete a user ID, you insert the following statement into skeleton IESUPDCF:

```
►►──DELete user_id──────────────────────────────────────────────────►◄
```

For the DELete statement, the user ID identifies the user which is to be deleted. It must be 4-8 alphanumeric characters long and may include the characters @, #, or $. Blanks are not allowed.

Skeleton IESUPDCF

The following figure shows skeleton IESUPDCF, shipped in ICCF library 59. Use this skeleton to ADD, ALTer, or DELete your user profiles.

```
* $$ JOB JNM=IESUPDCF,CLASS=0,DISP=D
* $$ PUN DISP=I,CLASS=0,PRI=9
// JOB IESUPDCF
// OPTION NOLOG
*
*  THIS SKELETON MAY BE USED BY THE ADMINISTRATOR TO GENERATE A
*  JOB FOR BATCH USER PROFILE MAINTENANCE.
*    1. IF THE CONTROL FILE BELONGS TO A CICS WITHOUT ICCF AND THIS
*       CICS DOES NOT SHARE THE CONTROL FILE WITH CICS/ICCF,
*       ADJUST THE '// DLBL' STATEMENT TO MAINTAIN
*       USER PROFILES IN THE RELATED CONTROL FILE.
*    2. SUPPLY AN OPERAND FOR THE ICCF PARAMETER, VALID OPERANDS ARE:
*         Yes    ... UPDATE USER PROFILES IN CONTROL FILE (CICS) AND
*                    IN THE DTSFILE (ICCF).
*         No     ... UPDATE USER PROFILES IN CONTROL FILE ONLY.
*                    INHIBIT CHANGES TO ICCF RELATED INFORMATION.
*         Ignore ... UPDATE USER PROFILES IN CONTROL FILE ONLY.
*                    THIS VALUE MUST BE USED IF THE CONTROL FILE
*                    IS USED IN CICS SUBSYSTEMS RUNNING WITHOUT ICCF.
*    3. INSERT THE ADD, ALTER AND DELETE STATEMENTS THAT YOU NEED TO
*       MAINTAIN USER PROFILES.
*          SAMPLE STATEMENTS:
*          ==================
*            * TEXT   ... A COMMENT LINE
*            ADd     USERID,PASSWD,PROFILE(,OPTIONAL PARAMETERS)
*            ALter   USERID(,OPTIONAL PARAMETERS)
*            Delete  USERID
*          EXPLANATION OF PARAMETERS:
*          ==========================
*           1. REQUIRED AND POSITIONAL PARAMETERS:
*           --------------------------------------
*            USERID  ... THE ID OF THE USER ( ADD, ALTER, DELETE )
*                        ( 4-8 CHARACTER / 4 CHARACTER FOR ICCF USER )
```

Figure 116. Skeleton IESUPDCF (Part 1 of 3)

```
*           PASSWD  ... THE PASSWORD OF THE USER ( ADD )
*                       ( 3-8 CHARACTERS )
*           PROFILE ... THE ID OF THE USER USED AS PROFILE FOR
*                       THE NEW USER  ( ADD )
*                       ( 4-8 CHARACTER / 4 CHARACTER FOR ICCF USER )
*          2. OPTIONAL PARAMETERS IN ADD/ALTER STATEMENT:
*          ---------------------------------------------
*           Catalog=   ... THE DEFAULT CATALOG OF THE USER
*                      EXAMPLE: CAT=VSESPUC
*           Days=      ... NUMBER OF DAYS IN EXPIRATION INTERVAL
*                      EXAMPLE: DAYS=20  ( RANGE: 0-365)
*           Library=   ... Primary ICCF library ( only ICCF users )
*                      EXAMPLE: LIB=20
*           Initial=   ... Initial function at SIGNON
*                      EXAMPLE: INIT=APPLNAME(A) ... FOR APPLICATION
*                               INIT=SELNAME(S)  ... FOR SELECTION P.
*           Natlang=   ... NATIONALLANGUAGE_INDICATOR
*                      EXAMPLE: NAT=E ( for English)
*           OID    =   ... OID CHARACTERS
*                      EXAMPLE: OID=ABC ( max. 20 Characters )
*           Operator=  OPERATOR ID
*                      EXAMPLE: OPER=OPE
*           PWD=       ... USER PASSWORD
*           PAssword=  EXAMPLE: PWD=PASSWD  ( 3-8 Characters )
*           PRiority=  ... OPERATOR PRIORITY
*                      EXAMPLE: PRIOR=5  ( RANGE: 0-255 )
*           Revoke=    ... REVOKE_DATE
*                      EXAMPLE: R=01/31/01 ( Format mm/dd/yy )
*           Synonym=   ... SYNONYMS MODEL
*                      EXAMPLE: SYNONYM=SYNS  ( 4-8 CHARACTERS )
*           Timeout=   ... TIMEOUT INTERVAL
*                      EXAMPLE: TIME=20  ( VALUES: 0,5,10,...,60 )
*           APM=Yes|No   ... APPLICATION PROFILE MAINTENANCE
*           BQA=Yes|No   ... MANAGE ALL BATCH QUEUES
*           CMD=Yes|No   ... ENTER CONSOLE COMMANDS
*           COU=Yes|No   ... FULL OUTPUT ON SYSTEM CONSOLE
*           COD=Yes|No   ... CONFIRM ON DELETE
*           DVF=Yes|No   ... DEFINE VSAM FILES
*           ESC=Yes|No   ... ESCAPE TO CICS
*           MVC=Yes|No   ... MANAGE VSAM CATALOGS
*           NEWS=Yes|No  ... DISPLAY NEWS TO USER
*           OLPD=Yes|No  ... DELETE OLPD INCIDENTS
*           PSL=Yes|No   ... OWNS A PRIVATE SUBLIBRARY
*           BAT=Yes|No   ... SUBMIT TO BATCH
*           SPM=Yes|No   ... SELECTION PANEL MAINTENANCE
*           UPM=Yes|No   ... USER PROFILE MAINTENANCE
*           XRF=Yes|No   ... XRF SIGNOFF
*           -                ... CONTINUATION CHARACTER
*    4. DELETE BLOCK 'UPDPL', IF YOU DO NOT WANT TO MAINTAIN
*       THE PRIMARY LIBRARY.
*
* ====> UPDATE NEXT LINE IF NECESSARY (SEE 1.)
// DLBL IESCNTL,'VSE.CONTROL.FILE',,VSAM,CAT=VSESPUC
*
// EXEC PROC=DTRICCF
// EXEC IESUPDCF,SIZE=64K
*
* ====> SUPPLY AN OPERAND FOR THE ICCF PARAMETER (SEE 2.)
ICCF=
*
* ====> INSERT STATEMENTS HERE (NO COMMENT '*' IN FIRST COLUMN, SEE 3.)
```

Figure 116. Skeleton IESUPDCF (Part 2 of 3)

```
* /*
// IF $RC=0 THEN
// GOTO STEP2
// IF $RC=4 THEN
// GOTO ERROR
// IF $RC>6 THEN
// GOTO END
// LOG
* ===> JOB 'DTRUPD' CREATED, ENSURE THAT THIS JOB IS EXECUTED NEXT
// NOLOG
// IF $RC=2 THEN
// GOTO STEP2
/. ERROR
// LOG
* ===> ERRORS IN INPUT DATA, STATEMENT(S) FLAGGED IN LISTING
// NOLOG
/. STEP2
*
* ====> DELETE BLOCK 'UPDPL', IF REQUIRED (SEE 4.)
* ************ BEGIN OF BLOCK 'UPDPL' ****************************
// EXEC PROC=IESUPDPL
* /*
* ************ END OF BLOCK 'UPDPL' ******************************
*
/. END
* /&
* $$ EOJ
```

Figure 116. Skeleton IESUPDCF (Part 3 of 3)

Using IESUPDCF

After making changes in skeleton IESUPDCF, submit the job for processing. Once the job is processed, check the output listing to see whether the job DTRUPD was created. Please note that this job will only be created when you specify ICCF=YES. If so:

- Check the system console, since job DTRUPD prompts you to disconnect the DTSFILE and waits for a response.
- Disconnect the DTSFILE (**/DISC DTSFILE**) and reply to the suspended job.
- Reconnect the DTSFILE after the job has terminated (**/CON DTSFILE**).
- If you have specified ICCF=YES, you will have two listings with the name IESUPDCF. (With ICCF=NO or ICCF=IGNORE you will get one IESUPDCF listing). Check both of them for flagged statements and return codes.

Return Codes Issued by IESUPDCF

0 No error. Job DTRUPD was not generated.

2 No error. Job DTRUPD was generated.

User action:
- Ensure that job (DTRUPD) is started immediately.
- Disconnect the DTSFILE when prompted on the system console.

4 The program has detected one or more invalid user statements in the job. The invalid statements are flagged in the listing. All valid statements are processed. Job DTRUPD was not generated.

User action:
- Examine the job listing.

- Correct the flagged job statements.
- Delete statements that are not flagged from the job, because they have been processed before.
- Submit the corrected job again.

6 The program has detected one or more invalid user statements in the job. The invalid statements are flagged in the listing. All valid statements are executed. Job DTRUPD was generated.

User action:
- Submit job DTRUPD.
- Examine the job listing.
- Correct the flagged job statements.
- Delete statements that are not flagged from the job, because they have been processed before.
- Submit the corrected job again.

8 The ICCF statements were ignored. ICCF=NO was specified in the job, but there was at least one statement that tried to alter an ICCF user definition. This statement was ignored.

User action:

If the erroneous statement is to be processed:
- Specify ICCF=YES.
- Delete all statements that are not flagged, because they have been processed before.
- Submit the corrected job again.

16 The program has been canceled due to severe errors.

User action:
- Examine the listing to determine the reason. The error might have been caused by one of the following:

  CDLOAD
  The program was unable to load the DTSFILE I/O routine DTSFILRT.

  CONTROL FILE
  A VSE/VSAM macro caused an error.

  GETVIS
  The partition GETVIS area is too small for the job.

Example of Completed Skeleton IESUPDCF

The following is an example of a completed IESUPDCF skeleton, which shows ADDing, ALTering, and DELeting users.

```
* $$ JOB JNM=IESUPDCF,CLASS=0,DISP=D
* $$ PUN DISP=I,CLASS=0,PRI=9
// JOB IESUPDCF
// OPTION NOLOG
*
*  THIS SKELETON MAY BE USED BY THE ADMINISTRATOR TO GENERATE A
*  JOB FOR BATCH USER PROFILE MAINTENANCE.
*       ...
*       ... Description is deleted.
*       ... (See skeleton IESUPDCF)
*
* ====> UPDATE NEXT LINE IF NECESSARY (SEE 1.)
// DLBL IESCNTL,'VSE.CONTROL.FILE',,VSAM,CAT=VSESPUC
*
// EXEC PROC=DTRICCF
// EXEC IESUPDCF,SIZE=64K
*
* ====> SUPPLY AN OPERAND FOR THE ICCF PARAMETER (SEE 2.)
ICCF=YES
*
* ====> INSERT STATEMENTS HERE (NO COMMENT '*' IN FIRST COLUMN, SEE 3.)
ADD NEWUSR,PASSWD,OLDUSR, -
    DAYS=30,TIMEOUT=15, -
    CPW=YES,PSL=YES
ALT MYUSER, PWD=NEWPWD, -
    DAYS=30,TIMEOUT=15, -
    CPW=YES,PSL=YES
DEL OLDUSR
/*
// IF $RC=0 THEN
// GOTO STEP2
// IF $RC=4 THEN
// GOTO ERROR
// IF $RC>6 THEN
// GOTO END
// LOG
* ===> JOB 'DTRUPD' CREATED, ENSURE THAT THIS JOB IS EXECUTED NEXT
// NOLOG
// IF $RC=2 THEN
// GOTO STEP2
/. ERROR
// LOG
* ===> ERRORS IN INPUT DATA, STATEMENT(S) FLAGGED IN LISTING
// NOLOG
/. STEP2
*
* ====> DELETE BLOCK 'UPDPL', IF REQUIRED (SEE 4.)
* ************ BEGIN OF BLOCK 'UPDPL' ****************************
// EXEC PROC=IESUPDPL
* /*
* ************ END OF BLOCK 'UPDPL' **********************************
/. END
/&
* $$ EOJ
```

Figure 117. Example of a Completed Skeleton IESUPDCF

Glossary

If you do not find the term you are looking for, refer to the index of this book or to the *IBM Glossary of Computing Terms* at:

`http://www.ibm.com/ibm/terminology`

The glossary includes definitions with:

- Symbol * where there is a one-to-one copy from the IBM Dictionary of Computing.
- Symbol (A) from the *American National Dictionary for Information Processing Systems* , copyright 1982 by the Computer and Business Equipment Manufacturers Association (CBEMA). Copies may be purchased from the American National Standards Institute, 1430 Broadway, New York, New York 10018. Definitions are identified by the symbol (A) after the definition.
- Symbols (I) or (T) from the *ISO Vocabulary - Information Processing* and the *ISO Vocabulary - Office Machines,* developed by the International Organization for Standardization, Technical Committee 97, Subcommittee 1. Definitions of published segments of the vocabularies are identified by the symbol (I) after the definition; definitions from draft international standards, draft proposals, and working papers in development by the ISO/TC97/SC1 vocabulary subcommittee are identified by the symbol (T) after the definition, indicating final agreement has not yet been reached among participating members.

access control. A function of VSE that ensures that the system and the data and programs stored in it can be accessed only by authorized users in authorized ways.

access control table (DTSECTAB). A table used by the system to verify a user's right to access a certain resource.

access method. A program, that is, a set of commands (macros), to define files or addresses and to move data to and from them; for example VSE/VSAM or VSE/VTAM.

ACF/VTAM. See VTAM.

address space. A range of up to two gigabytes of contiguous virtual storage addresses that the system creates for a user. Unlike a data space, an address space contains user data **and** programs, as well as system data and programs, some of which are common to all address spaces. Instructions execute in an address space (not a data space). Contrast with data space.

Advanced Function Printing (AFP). A group of IBM licensed programs that support APA printers.

*** alternate index.** In systems with VSE/VSAM, a collection of index entries related to a given base cluster and organized by an alternate key, that is, a key other than the prime key of the base cluster data records; it gives an alternate directory for finding records in the data component of a base cluster. See also *path*.

*** alternate library.** An interactively accessible library that can be accessed from a terminal when the user of that terminal issues a connect or switch library request. Synonymous with *public library*.

*** application profile.** A control block in which the system stores the characteristics of one or more application programs.

ASI (automated system initialization) procedure. A set of control statements which specifies values for an automatic system initialization.

*** assemble.** To translate an assembly language program into an object program. (T)

*** assembler.** A computer program that converts assembly language instructions into object code.

assembler language. A programming language whose instructions are usually in one-to-one correspondence with machine instructions and allows to write macros.

*** automated system initialization (ASI).** A function that allows control information for system startup to be cataloged for automatic retrieval during system startup.

*** autostart.** A facility that starts up VSE/POWER with little or no operator involvement.

background partition. An area of virtual storage in which programs are executed under control of the system. By default, the partition has a processing priority lower than any of the existing foreground partitions.

block. Usually, a block consists of several records of a file that are transmitted as a unit. But if records are very large, a block can also be part of a record only. On an FBA disk, a block is a string of 512 bytes of data. See also *control block*.

*** catalog.** 1. A directory of files and libraries, with reference to their locations. A catalog may contain other information such as the types of devices in which the files are stored, passwords, blocking factors. (I) (A) 2. To store a library member such as a phase, module, or book in a sublibrary.

See also *VSAM master catalog, VSAM user catalog*.

* **cataloged procedure.** A set of control statements placed in a library and retrievable by name.

CICS Transaction Server for VSE/ESA. This is the successor system to CICS/VSE.

* **CICS/VSE.** Customer Information Control System/VSE.

* **common library.** An interactively accessible library that can be accessed by any user of the system or subsystem that owns the library.

* **compaction.** In SNA, the transformation of data by packing two characters in a byte so as to take advantage of the fact that only a subset of the allowable 256 characters is used; the most frequently sent characters are compacted.

* **compile.** To translate a source program into an executable program (an object program). See also *assembler*.

conditional job control. The capability of the job control program to process or to skip one or more statements based on a condition that is tested by the program.

* **configuration.** The devices and programs that make up a system, subsystem, or network.

connect. To authorize library access on the lowest level. A modifier such as "read" or "write" is required for the specified use of a sublibrary.

* **control area (CA).** In VSE/VSAM, a group of control intervals used as a unit for formatting a data set before adding records to it. Also, in a key-sequenced data set, the set of control intervals, pointed to by a sequence-set index record, that is used by VSAM for distributing free space and for placing a sequence-set index record adjacent to its data.

control block. An area within a program or a routine defined for the purpose of storing and maintaining control information.

* **control interval (CI).** A fixed-length area of disk storage where VSE/VSAM stores records and distributes free space. It is the unit of information that VSE/VSAM transfers to or from disk storage. For FBA, it must be an integral multiple, to be defined at cluster definition, of the block size.

DASD sharing. An option that lets independent computer systems use common data on shared disk devices.

* **data entry panel.** A panel in which the user communicates with the system by filling in one or more fields. See also *panel* and *selection panel*.

data file. See *file*.

Data Interfile Transfer, Testing and Operations (DITTO) utility. An IBM program that provides file-to-file services for card I/O, tape, and disk devices. The latest version is called DITTO/ESA for VSE.

data set. See *file*.

default value. A value assumed by the program when no value has been specified by the user.

* **device address.** 1. The identification of an input/output device by its channel and unit number. 2. In data communication, the identification of any device to which data can be sent or from which data can be received.

* **device type code.** The four- or five-digit code to be used for defining an I/O device to a computer system.

* **dialog.** 1. In an interactive system, a series of related inquiries and responses similar to a conversation between two people. 2. For z/VSE, a set of panels that can be used to complete a specific task; for example, defining a file.

direct access. Accessing data on a storage device using their address and not their sequence. This is the typical access on disk devices as opposed to magnetic tapes. Contrast with *sequential access*.

directory. 1. A table of identifiers and references to the corresponding items of data. (I) (A) 2. In VSE, specifically, the index for the program libraries. See also *library directory* and *sublibrary directory*.

disk operating system residence volume (DOSRES). The disk volume on which the system sublibrary IJSYSRS.SYSLIB is located including the programs and procedures required for system startup.

disposition. A means of indicating to VSE/POWER how job input and output is to be handled. A job may, for example, be deleted or kept after processing.

DOSRES. Disk operating system residence volume.

dummy device. A device address with no real I/O device behind it. Input and output for that device address are spooled on disk.

* **dump.** 1. Data that has been dumped. (T) 2. To record, at a particular instant, the contents of all or part of one storage device in another storage device. Dumping is usually for the purpose of debugging. (T)

dynamic class table. Defines the characteristics of dynamic partitions.

dynamic partition. A partition created and activated on an 'as needed' basis that does not use fixed static allocations. After processing, the occupied space is released. Contrast with *static partition*.

Enterprise Systems Architecture (ESA). See *ESA/390*.

ESA/390. IBM Enterprise Systems Architecture/390. The latest extension to the IBM System/370 architecture which includes the advanced addressability feature and advanced channel architecture.

* **escape.** To return to the original level of a user interface.

extent. Continuous space on a disk or diskette occupied by or reserved for a particular file or VSAM data space.

fast service upgrade (FSU). A service function of z/VSE for the installation of a refresh release without regenerating control information such as library control tables.

FBA disk device. Fixed-block architecture disk device.

* **file.** A named set of records stored or processed as a unit. (T) Synonymous with *data set*.

* **fragmentation (of storage).** In virtual system, inability to assign real storage locations to virtual storage addresses because the available spaces are smaller than the page size.

FULIST (FUnction LIST). A type of selection panel that displays a set of files and/or functions for the choice of the user.

* **generate.** To produce a computer program by selecting subsets of skeletal code under the control of parameters. (A)

generation. See *macro generation*.

* **GETVIS space.** Storage space within a partition or the shared virtual area, available for dynamic allocation to programs.

hardcopy file. A system file on disk, used to log all lines of communication between the system and the operator at the system console, to be printed on request.

* **hardware.** All or part of the physical components of an information processing system, such as computers or peripheral devices. (T) (A) Contrast with *software*.

* **help panel.** A display of information provided by the system in response to a user's help request.

host transfer file (HTF). Used by the Workstation File Transfer Support of z/VSE as an intermediate storage area for files that are sent to and from IBM Personal Computers.

* **initial program load (IPL).** The process of loading system programs and preparing the system to run jobs.

interactive. A characteristic of a program or system that alternately accepts input and then responds. An interactive system is conversational, that is, a continuous dialog exists between user and system. Contrast with *batch*.

interactive interface. A system facility which controls how different users see and work with the system by means of user profiles. When signing on, the interactive interface makes available those parts of the system authorized by the profile. The interactive interface has sets of selection- and data-entry panels through which users communicate with the system.

interactive partition. An area of virtual storage for the purpose of processing a job that was submitted interactively via VSE/ICCF.

interface. A shared boundary between two hardware or software units, defined by common functional or physical characteristics. It might be a hardware component or a portion of storage or registers accessed by several computer programs.

job accounting. A system function that lists how much every job step uses of the different system resources.

* **job catalog.** A catalog made available for a job by means of the filename IJSYSUC in the respective DLBL job control statement.

job control language (JCL). A language that serves to prepare a job or each job step of a job to be run. Some of its functions are: to identify the job, to determine the I/O devices to be used, set switches for program use, log (or print) its own statements, and fetch the first phase of each job step.

job step. One of a group of related programs complete with the JCL statements necessary for a particular run. Every job step is identified in the job stream by an EXEC statement under one JOB statement for the whole job.

job stream. The sequence of jobs as submitted to an operating system.

label information area. An area on a disk to store label information read from job control statements or commands. Synonymous with *label area*.

* **librarian.** The set of programs that maintains, services, and organizes the system and private libraries.

library. See *VSE library* and *VSE/ICCF library*.

* **library directory.** The index that enables the system to locate a certain sublibrary of the accessed library.

*** library member.** The smallest unit of data that can be stored in and retrieved from a sublibrary.

*** licensed program.** A separately priced program and its associated materials that bear an IBM copyright and are offered to customers under the terms and conditions of the IBM Customer Agreement (ICA).

*** linkage editor.** A program used to create a phase (executable code) from one or more independently translated object modules, from one or more existing phases, or from both. In creating the phase, the linkage editor resolves cross references among the modules and phases available as input. The program can catalog the newly built phases.

*** local address.** In SNA, an address used in a peripheral node in place of a network address and transformed to or from a network address by the boundary function in a subarea node.

*** lock file.** In a shared disk environment under VSE, a system file on disk used by the sharing systems to control their access to shared data.

*** logging.** The recording of data about specific events.

logical record. A user record, normally pertaining to a single subject and processed by data management as a unit. Contrast with *physical record* which may be larger or smaller.

logical unit (LU). 1. A name used in programming to represent an I/O device address. 2. In SNA, a port through which a user accesses the SNA network, to communicate with another user.

master console. In z/VSE, one or more consoles that receive all system messages, except for those that are directed to a particular console. The operator of a master console can reply to all outstanding messages and enter all system commands.

major node. In VTAM, a set of minor nodes that can be activated as a group. See *node* and *minor node*.

*** member.** The smallest unit of data that can be stored in and retrieved from a sublibrary. See also *library member*.

message. 1. In VSE, a communication sent from a program to the operator or user. It can appear on a console, a display terminal or on a printout. 2. In telecommunication, a logical set of data being transmitted from one node to another.

*** minor node.** In VTAM programs, a uniquely-defined resource within a major node. See *node* and *major node*.

*** module.** A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading; for example, the input to, or output from an assembler, compiler, linkage editor, or executive routine. (A)

*** network address.** In SNA, an address, consisting of subarea and element fields, that identifies a link, link station, or NAU. Subarea nodes use network addresses; peripheral nodes use local addresses. The boundary function in the subarea node to which a peripheral node is attached transforms local addresses to network addresses and vice versa. See *local address*. See also *network name*.

network addressable unit (NAU). In SNA, a logical unit, a physical unit, or a system services control point. It is the origin or the destination of information transmitted by the path control network. Each NAU has a network address that represents it to the path control network. See also *network name*, *network address*.

networking. Making use of the services of a network program.

network name. 1. In SNA, the symbolic identifier by which users refer to a NAU, link, or link station. See also *network address*. 2. In a multiple-domain network, the name of the APPL statement defining a VTAM application program. This is its network name which must be unique across domains.

*** node.** 1. In SNA, an end point of a link or junction common to two or more links in a network. Nodes can be distributed to host processors, communication controllers, cluster controllers, or terminals. Nodes can vary in routing and other functional capabilities. 2. In VTAM programs, a point in a network defined by a symbolic name. Synonymous with *network node*. See *major node* and *minor node*.

nonprogrammable workstation (NPWS). A workstation that does not have processing capability and that does not allow the user to change its functions. Contrast with *programmable workstation*.

object module (program). A program unit that is the output of an assembler or compiler and is input to a linkage editor.

optional program. An IBM licensed program that a user can install on VSE by way of available installation-assist support.

page data set (PDS). One or more extents of disk storage in which pages are stored when they are not needed in processor storage.

page frame. An area of processor storage that can contain a page.

panel. The complete set of information shown in a single display on a terminal screen. Scrolling back and forth through panels is like turning manual pages. See also *selection panel* and *data entry panel*.

partition. A division of the virtual address area available for running programs. See also *dynamic partition*, *static partition*.

* **path.** 1. In VTAM, the intervening nodes and data links connecting a terminal and an application program in the host processor. 2. In VSAM, a named logical entity providing access to the records of a base cluster either directly or through an alternate index.

personal computer (PC). A microcomputer for individuals or small businesses.

* **physical record.** The amount of data transferred to or from auxiliary storage. Synonymous with *block*.

* **physical unit (PU).** In SNA, the component that manages and monitors the resources of a node.

* **primary library.** A VSE library owned and directly accessible by a certain terminal user.

priority. A rank assigned to a partition or a task that determines its precedence in receiving system resources.

private area. The part of an address space that is available for the allocation of private partitions. Its maximum size can be defined during IPL. Contrast with *shared area*.

procedure. See *cataloged procedure*.

* **processing.** The performance of logical operations and calculations on data, including the temporary retention of data in processor storage while this data is being operated upon.

* **processor.** In a computer, a functional unit that interprets and executes instructions. A processor consists of at least an instruction control unit and an arithmetic and logic unit. (T)

profile. A description of the characteristics of a user or a computer resource.

* **programmable workstation.** A workstation that has some degree of processing capability and that allows the user to change its functions. Contrast with *nonprogrammable workstation*.

prompt. To issue messages to a terminal or console user, requesting information necessary to continue processing.

protocol. In SNA, the set of rules for requests and responses between communicating nodes that want to exchange data.

* **queue file.** A direct access file maintained by VSE/POWER that holds control information for the spooling of job input and job output.

record. A set of related data or words, treated as a unit. See *logical record*, *physical record*.

* **remote job entry (RJE).** Submission of jobs through an input unit that has access to a computer through a data link.

* **restore.** To write back onto disk data that was previously written from disk onto an intermediate storage medium such as tape.

* **routing.** The assignment of the path by which a message will reach its destination.

SAM ESDS file. A SAM file managed in VSE/VSAM space, so it can be accessed by both SAM and VSE/VSAM macros.

* **search chain.** The order in which chained sublibraries are searched for the retrieval of a certain library member of a specified type.

security. See *access control*.

* **selection panel.** A displayed list of items from which a user can make a selection. Synonymous with *menu*.

sequential access. The serial retrieval of records in their entry sequence or serial storage of records with or without a premeditated order. Contrast with *direct access*.

* **sequential file.** A file in which records are processed in the order in which they are entered and stored.

service node. Within the VSE unattended node support, a processor used to install and test a master VSE system which is copied for distribution to the unattended nodes. Also, program fixes are first applied at the service node and then sent to the unattended nodes.

* **service program.** A computer program that performs functions in support of the system. Synonymous with *utility program*.

shared area. An area of storage that is common to all address spaces in the system. z/VSE has two shared areas:
1. The shared area (24 bit) is allocated at the start of the address space and contains the supervisor, the SVA (for system programs and the system GETVIS area), and the shared partitions.
2. The shared area (31 bit) is allocated at the end of the address space and contains the SVA (31 bit) for system programs and the system GETVIS area.

* **shared virtual area (SVA).** A high address area that contains a system directory list (SDL) of frequently used phases, resident programs that can be shared between partitions, and an area for system support.

*** skeleton.** A set of control statements, instructions, or both, that requires user-specific information to be inserted before it can be submitted for processing.

*** software.** All or part of the programs, procedures, rules, and associated documentation of a data processing system. Software is an intellectual creation that is independent of the medium on which it is recorded. (T)

*** spooling.** The use of disk storage as buffer storage to reduce processing delays when transferring data between peripheral equipment and the processors of a computer. In VSE, this is done under the control of VSE/POWER.

*** standard label.** A fixed-format record that identifies a volume of data such as a tape reel or a file that is part of a volume of data.

startup. The process of performing IPL of the operating system and of getting all subsystems and application programs ready for operation.

static partition. A partition, defined at IPL time and occupying a defined amount of virtual storage that remains constant. Contrast with *dynamic partition*.

storage dump. See *dump*.

*** subarea.** A portion of the SNA network consisting of a subarea node, attached peripheral nodes, and associated resources. Within a subarea node, all links, and adjacent link stations in attached peripheral or subarea nodes that are addressable within the subarea share a common subarea address and have distinct element addresses.

sublibrary. A subdivision of a library. Members can only be accessed in a sublibrary.

sublibrary directory. An index for the system to locate a member in the accessed sublibrary.

submit. A VSE/POWER function that passes a job to the system for processing.

*** subsystem.** A secondary or subordinate system, usually capable of operating independently of, or asynchronously with, a controlling system. (T)

*** supervisor.** The part of a control program that coordinates the use of resources and maintains the flow of processor operations.

supervisor mode. See *ESA mode*.

SYSRES. System residence file.

*** system console.** A console, usually equipped with a keyboard and display screen for control and communication with the system.

system directory list (SDL). A list containing directory entries of frequently-used phases and of all phases resident in the SVA. The list resides in the SVA.

*** system file.** A file used by the operating system, for example, the hardcopy file, the recorder file, the page data set.

*** system recorder file.** The file used to record hardware reliability data. Synonymous with *recorder file*.

system residence file (SYSRES). The z/VSE system sublibrary IJSYSRS.SYSLIB that contains the operating system. It is stored on the system residence volume DOSRES.

*** tailor.** A process that defines or modifies the characteristics of the system.

*** terminal.** A point in a system or network at which data can either enter or leave. (A) Usually a display screen with a keyboard.

terminal control table (TCT). A control block in which the system stores information about the characteristics and modes of operation of the terminals defined to the system.

*** transaction.** 1. In a batch or remote batch entry, a job or job step. 2. In CICS/VSE, one or more application programs that can be used by a display station operator. A given transaction can be used concurrently from one or more display stations. The execution of a transaction for a certain operator is also referred to as a task. A given task can relate only to one operator.

*** transmit.** To send data from one place for reception elsewhere. (A)

unattended node support. A set of functions allowing one or more VSE systems to run without an operator being present. The systems are connected to a single central host.

*** utility program.** 1. A computer program in general support of computer processes; for example, a diagnostic program, a trace program, or a sort program. (T) Synonymous with *service program*. 2. A program designed to perform an everyday task such as copying data from one storage device to another. (A)

virtual address. An address that refers to a location in virtual storage. It is translated by the system to a processor storage address when the information stored at the virtual address is to be used.

*** virtual address space.** A subdivision of the virtual address area available to the user for the allocation of private, nonshared partitions.

Virtual Machine/Enterprise Systems Architecture (VM/ESA). The most advanced VM system currently available.

virtual storage. Addressable space image for the user from which instructions and data are mapped into processor (real) storage locations.

volume. A data carrier that is mounted and demounted as a unit, for example, a reel of tape or a disk pack. (I) Some disk units have no demountable packs. In that case, a volume is the portion available to one read/write mechanism.

volume ID. The volume serial number, which is a number in a volume label assigned when a volume is prepared for use by the system.

VSE (Virtual Storage Extended). A system that consists of a basic operating system and any IBM supplied and user-written programs required to meet the data processing needs of a user. VSE and the hardware it controls form a complete computing system. Its current version is called z/VSE.

VSE/ICCF (VSE/Interactive Computing and Control Facility). An IBM program that serves as interface, on a time-slice basis, to authorized users of terminals linked to the system's processor.

VSE/ICCF library. A file composed of smaller files (libraries) including system and user data which can be accessed under the control of VSE/ICCF.

VSE library. A collection of programs in various forms and storage dumps stored on disk. The form of a program is indicated by its member type such as source code, object module, phase, or procedure. A VSE library consists of at least one sublibrary which can contain any type of member.

VSE/POWER. An IBM program primarily used to spool input and output. The program's networking functions enable a VSE system to exchange files with or run jobs on another remote processor.

VSE/VSAM (VSE/Virtual Storage Access Method). An IBM access method for direct or sequential processing of fixed and variable length records on disk devices.

VSE/VSAM catalog. A file containing extensive file and volume information that VSE/VSAM requires to locate files, to allocate and deallocate storage space, to verify the authorization of a program or an operator to gain access to a file, and to accumulate use statistics for files.

VTAM (Virtual Telecommunications Access Method). An IBM program which controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability; it supports application programs and subsystems (VSE/POWER, for example).

workstation. See *programmable* and *nonprogrammable workstation*.

Workstation File Transfer Support. Enables the exchange of data between IBM Personal Computers (PCs) linked to a z/VSE host system where the data is kept in intermediate storage. PC users can retrieve that data and work with it independently of z/VSE.

z/VM. z/Virtual Machine.

z/VSE (z/Virtual Storage Extended). The most advanced VSE system currently available.

31-bit addressing. Provides addressability for address spaces of up to 2 gigabytes. (The maximum amount of addressable storage in previous systems was 16 megabytes.)

Index

Special characters

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

Readers' Comments — We'd Like to Hear from You

IBM z/VSE
Administration
Version 3 Release 1

Publication No. SC33-8224-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name

Address

Company or Organization

Phone No.

Readers' Comments — We'd Like to Hear from You
SC33-8224-00

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Deutschland Entwicklung GmbH
Department 3248
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany

Fold and Tape **Please do not staple** Fold and Tape

SC33-8224-00

Cut or Fold
Along Line

File Number: S370/S390-34
Program Number: 5609-ZVS

Printed in USA

SC33-8224-00

Spine information:

z/VSE Administration Version 3 Release 1 SC33-8224-00