IBM z/VSE
VSE Central Functions
8.1

VSE/ICCF
Administration and Operation

SC33-8329-01

Note!

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

First Edition (March 2007)

This edition applies to Version 8 Release 1 of the IBM Virtual Storage Extended/Interactive Computing and Control Facility (VSE/ICCF), which is part of VSE Central Functions, Program Number 5686-CF8, 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 3282
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
```

Contents

Figures

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.

About This Publication

This publication is intended for personnel concerned with the administration and operation of the IBM® Virtual Storage Extended/Interactive Computing and Control Facility (VSE/ICCF for short).

VSE/ICCF is part of IBM z/Virtual Storage Extended (z/VSE).

VSE/ICCF as delivered to you is operational once z/VSE is installed. This VSE/ICCF meets the requirements of many computing environments. In other cases, VSE/ICCF may have to be tailored to individual requirements. It is these necessities that the manual primarily addresses.

How This Publication is Organized

The publication presents its information in the following sequence:

- Chapter 1, "Introduction," on page 1 presents the operational concepts of VSE/ICCF and gives information about the scope of the available support.
- Chapter 2, "Tailoring VSE/ICCF," on page 23 describes how you can tailor the VSE/ICCF that is delivered and installed with z/VSE. This includes aspects such as access control, performance, and storage requirements.
- Chapter 3, "Operation," on page 59 describes the procedures needed to start up and operate VSE/ICCF.
- Chapter 5, "Utility Programs," on page 91 describes the available background control programs and the utility programs to be used for operating and maintaining VSE/ICCF.

Type codes used in this publication to refer to terminal devices or other data processing machines are the type codes of IBM devices and machines.

Where to Find More Information

The VSE/ICCF User's Guide provides further details on VSE/ICCF.

z/VSE IBM Documentation

IBM Documentation is the new home for IBM's technical information. The z/VSE IBM Documentation can be found here:

https://www.ibm.com/docs/en/zvse/6.2

You can also find VSE user examples (in zipped format) at

https://public.dhe.ibm.com/eserver/zseries/zos/vse/pdf3/zVSE_Samples.pdf

Summary of Changes

This dokumentation documents Version 8 Release 1 of VSE/ICCF which is part of z/VSE.

Changes for VSE/ICCF 8.1

Note, that the name VSE/ESA has changed to z/VSE. However, the names of many features and programs related to z/VSE remain unchanged (such as, for example, IBM COBOL for VSE/ESA, or TCP/IP for VSE/ESA).

Language Environment

The Language Environment LE/VSE and the following compilers are not supported in a VSE/ICCF interactive partition:

- COBOL for VSE/ESA
- PL/I for VSE/ESA
- C for VSE/ESA

It is recommended to use dynamic partitions instead.

Security

z/VSE supports two security components; either the Basic Security Manager (BSM) or the External Security Manager (ESM). Both components keep and maintain user IDs and passwords on their own. But VSE/ICCF still maintains user IDs and passwords in its DTSFILE.

When a user signs on to VSE/ICCF, the following applies:

- Logon from the Interactive Interface is always allowed.
- Native logon to VSE/ICCF requires either the password from the DTSFILE or the central password from the BSM or ESM.

DTSUTIL Count Operand

See “DSERV Command” on page 151.

Chapter 1. Introduction

This topic introduces some basic concepts of the IBM Virtual Storage Extended/Interactive Computing and Control Facility (VSE/ICCF). The section also gives a short description of VSE/ICCF's main functions.

Concepts

Basically, VSE/ICCF consists of

- A main control program
- Command processors
- Interactive partitions where terminal users' jobs are executed
- Background control programs
- A library on a direct access device for the storage of users' data and programs.

In addition, utility programs are provided to assist the user in doing his job more effectively.

These VSE/ICCF components are discussed in the following sections.

Main Control Program

The main control program supervises the activities of all VSE/ICCF components, except terminal I/O operations. Its responsibilities include attaching and monitoring subtasks, scheduling jobs, intercepting SVCs, performing library I/O, and allocating buffers. Terminal operations are carried out by the terminal control program of CICS (Customer Information Control System). CICS is a *base program* of z/VSE.

Command Processors

The command processors read and interpret commands and data from the terminal and carry out requested functions such as saving members in libraries, printing a member on the terminal, or entering a line of input.

Terminal Control Program

Terminal I/O in VSE/ICCF is handled by CICS. CICS has its own VSE partition where VSE/ICCF occupies a portion of the CICS GETVIS area.

Whenever this documentation uses the term *terminal control program*, or *TCP* for short, it is referring to CICS.

Background Control Programs

These programs provide facilities for writing and executing procedures (DTSPROCS), for submitting jobs to a VSE/POWER controlled partition (DTSSUBMT), and for linking programs for execution in an interactive partition (LINKNGO).

"DTSPROCS" and "LINKNGO" are described in the VSE/ICCF User's Guide For DTSSUBMT refer to Chapter 5, “Utility Programs,” on page 91.

z/VSE provides its own Interactive Interface which further eases working with the above facilities. This includes a set of dialogs for program development and for submitting jobs to VSE/POWER.

Utility Programs

VSE/ICCF also provides a set of utility programs for both the VSE/ICCF administrator and the terminal user. These utilities perform frequently used, standardized functions like sorting, punching and loading etc. They are described in Chapter 5, “Utility Programs,” on page 91.

Interactive Partitions

When the terminal user presents a job for execution in the VSE/ICCF partition (via an /EXEC, /RUN or /ENDRUN command), the system attempts to find a block of virtual storage into which to schedule the job. This block of virtual storage is called an interactive partition. Interactive partitions have characteristics similar to VSE partitions: they have their own communication region, their own storage protect key, and their own GETVIS area.

Interactive partitions may be associated with up to four classes, which may be used to direct certain kinds of jobs to certain partitions. For example, some partitions may have pre-allocated work areas, while others may not. Or, some interactive partitions may be of a standard size, while others may be larger to accommodate special programs. The VSE/ICCF system that is delivered to you has preset values for the size, number, and other characteristics of interactive partitions. You may change these values during the tailoring of VSE/ICCF (with options PARTN/PARTX of the tailoring macro DTSOPTNS). VSE/ICCF can support up to 35 interactive partitions.

Time-Sliced Execution

Once a job has been scheduled into an interactive partition it competes for task resources with other active processing requests on a time-sliced basis. When the job begins execution, it continues to run until one of the following occurs:

- A conversational read is encountered. This causes job execution to be suspended until the conversational input has been entered, after which the job is again made eligible for execution.
- The print buffer area is full or a /FORCE command has been given. This causes the job to be suspended until all the print lines have been transferred to the terminal. Then the job is made eligible for execution again.
- The interactive partition's time slice elapses. This causes the execution to be suspended until a new time slice becomes available.
- The job terminates. At this point, the remaining output is transferred to the terminal and the interactive partition is made available to another job.

Foreground versus Background

The terms 'Foreground' and 'Background' as they are used in connection with VSE/ICCF refer to the VSE/ICCF environment and should not be confused with the VSE foreground and background partitions. All VSE/ICCF system, full-screen editor, and context editor commands are executed in what is called a foreground task, whereas all compilations and program executions are performed as a background task. The terminal control program supervises the execution of all foreground tasks (these are CICS* transactions). Background tasks (which are VSE subtasks) run in interactive partitions.

The commands that are processed in the foreground have a higher priority than the functions performed in the background. This serves to maintain an adequate level of response to terminal users who are doing command work, which requires very little of the system's computing resources.

Background processing is concerned with the compilation and execution of programs in interactive partitions. Background execution requests are initiated whenever you issue the /RUN, /EXEC or /ENDRUN commands.

The Command Language

The various commands and job statements which make up the VSE/ICCF command language fall into the following categories:

- System Commands – System commands direct general system functions; for example, the /COPY and the /LIB commands.
- Full Screen Editor Commands – The full-screen editor commands generally relate to the scanning or manipulation of data in your input area or in a library member. The term 'full screen' refers to the fact that nearly the entire screen is available for entering data or commands. Editor commands are used

to control what is displayed on the screen and how it is displayed. They also relate to moving the current line pointer, locating areas within the file or changing or adding data within the current line. A subgroup of the editor commands, the editor line commands, enables you to change individual or multiple individual lines.

- Context Editor Commands – The context editor is used for the same purposes as the full-screen editor. Generally, the context editor commands are the same as the full screen editor commands. However, there are some editor commands, such as REWRITE or VERIFY, which are only useful in context editing. On the other hand, some editor commands, such as LINEMODE and REPLACE, yield different results for the two editors. Like system commands, context editor commands have an immediate effect on the data they refer to.
- Job Entry Statements – Job entry statements direct and control the execution of jobs. They look very much like system commands except that a system command is put into effect immediately, whereas job entry statements are placed in job streams and the functions that they request are not carried out until the job is actually run.
- Dump Commands – The dump commands enable you to display information from programs that have ended abnormally, provided you have specified the DUMP option in your /OPTION statement.
- Procedural and Macro Commands – A procedure or macro is a VSE/ICCF library member that contains executable commands, job entry statements, data, and procedure processor orders or macro orders. Both macros and procedures are invoked by specifying the member name as a command, for example SUBMIT. If a macro is invoked in an edit session, it must be preceded by an @; for example: @COPY. All statements and commands contained in the specified member are then executed. A macro runs as a foreground task under control of the terminal control program. A procedure runs as a background task in an interactive partition.

Terminal Facilities

- Screen Flexibility – As active portion of the screen you can set any contiguous rows or columns. For example, you can equate PF key 1 to a command which sets the active area to lines 15 to 24. PF key 2 can be set to a command which sets the active area to lines 4 to 14. Then, you can display data in one area, press a program function key, and display data in the other area. This facility allows you to retain data in one area of the screen while actively using another area. You can also set the active screen area based on columns instead of rows if the application warrants it. This facility is most useful in command mode.

  The full-screen editor offers, in addition, a split-screen facility. The user can split the physical screen into up to 8 logical screens in each of which he can edit another member. Each logical screen has its own command area and a scale line. Moreover, a logical screen can be split into up to 8 format areas in each of which the user can edit different parts of a member independent from each other. Each format area has its own command area, but no scale line.
- Multiple Line Input – Several commands or data lines can be processed for a single 'ENTER key' response by separating the logical lines with end of line characters. If you are using an IBM 3270 Information Display System, you can specify up to four lines (256 characters) for input.
- Program Function Keys – A program function key can be equated to any command or data line. It can be equated to a partial command, including a parameter. This parameter is replaced by data that you type in just before you press the PF key. If you do not have PF keys, you can access PF key functions via the /PFn command.

  Each of the following modes has its own complete independent set of PF keys (PF1 through PF24):

  Command mode
  Edit mode
  List mode (LS/SP)
  Execution and conversational read (EX/RD).
- Uppercase/Lowercase Facilities – You can switch your terminal between uppercase and mixed (upper-/lower-) case processing mode for text applications.

- Control Character Options – You can assign a certain control character function to a key on your keyboard. The following control characters are supported: logical line end, escape character, hex character, delete character, tab character and backspace character.

Formats of commands and statements described in this publication follow the notational rules used in the VSE/ICCF User's Guide.

Hardcopy Facilities

If you are using a display screen you may direct your output to any available IBM 328x printer. The data to be printed is queued on disk and automatically printed, leaving your terminal free for other functions while printing is taking place. In addition, you may not only direct your printed output to a queue associated with a printer, you may also direct your hardcopy output to a private queue or destination and then direct it to be printed at some later time on a given printer.

LOGON Procedure

Your user profile may indicate that a macro or a procedure can be executed at logon time. Such a macro or procedure may be used to set defaults, display your mail or any identifier or special instructions on the terminal, or to carry out any other function related to your logon.

Broadcast Records

Up to 8 broadcast records may be entered into the VSE/ICCF library file. These broadcast records are displayed on the user terminal when the user logs on to VSE/ICCF.

The Library File

The VSE/ICCF library file (file name DTSFILE) is a file on a DASD device. This file may be distributed over one or more extents on one or more DASD spindles but it may not be shared between CPUs. If the file is a multivolume file, all disks must be of the same device type. Access to the data in this file is via a specialized VSE/ICCF direct access method. The records are blocked and 88 bytes in length. The library file is divided into three logical areas:

- Fixed area: containing system record, user profile records, and library header records. The size of this area is defined once and may not be redefined without rebuilding the library file.
- Permanent area: this area is distributed among the various VSE/ICCF users in the form of user libraries and their corresponding directories. The size of a user library is not fixed, it competes for records with other user libraries. The records in one user library need not be contiguous, they may be intermixed with the records of another user library. Free records of the VSE/ICCF library file are taken from two sets of records: the *high file* records and the *free chain* records. The *high file* area starts next to the non-free record with the highest record number and extends up to the end of the file. The *free chain* records are scattered across the VSE/ICCF library file. The free chain contains the records which the user freed; by purging a member, for example. When free records are needed (for example, to create a new member), VSE/ICCF first tries to get the records from the high file area. If there are no more records in the high file area, the records are taken from the free chain. The directory records are chained from the library header records. The free chain as well as the broadcast records within the permanent area are chained from the system record.
- Temporary area: this area competes with the permanent area for records. The records in the temporary area are chained from VSE/ICCF internal control blocks. The temporary space is subdivided into four subpools:

  Input area
  Punch area
  Print area
  Log area

  The input area is used by foreground commands to build and edit members. In background execution it is used as SYSIPT.

The punch area is used by background executions for SYSPCH output. It is also used as stack area for the editor.

The print area is used by background executions for SYSLOG and SYSLST output.

The log area is used for logging of commands and command replies.

One set of temporary areas is assigned to each user. A temporary area is assigned the first time it is used and then remains assigned until logoff.

The following illustration shows the logical structure of the VSE/ICCF library file.

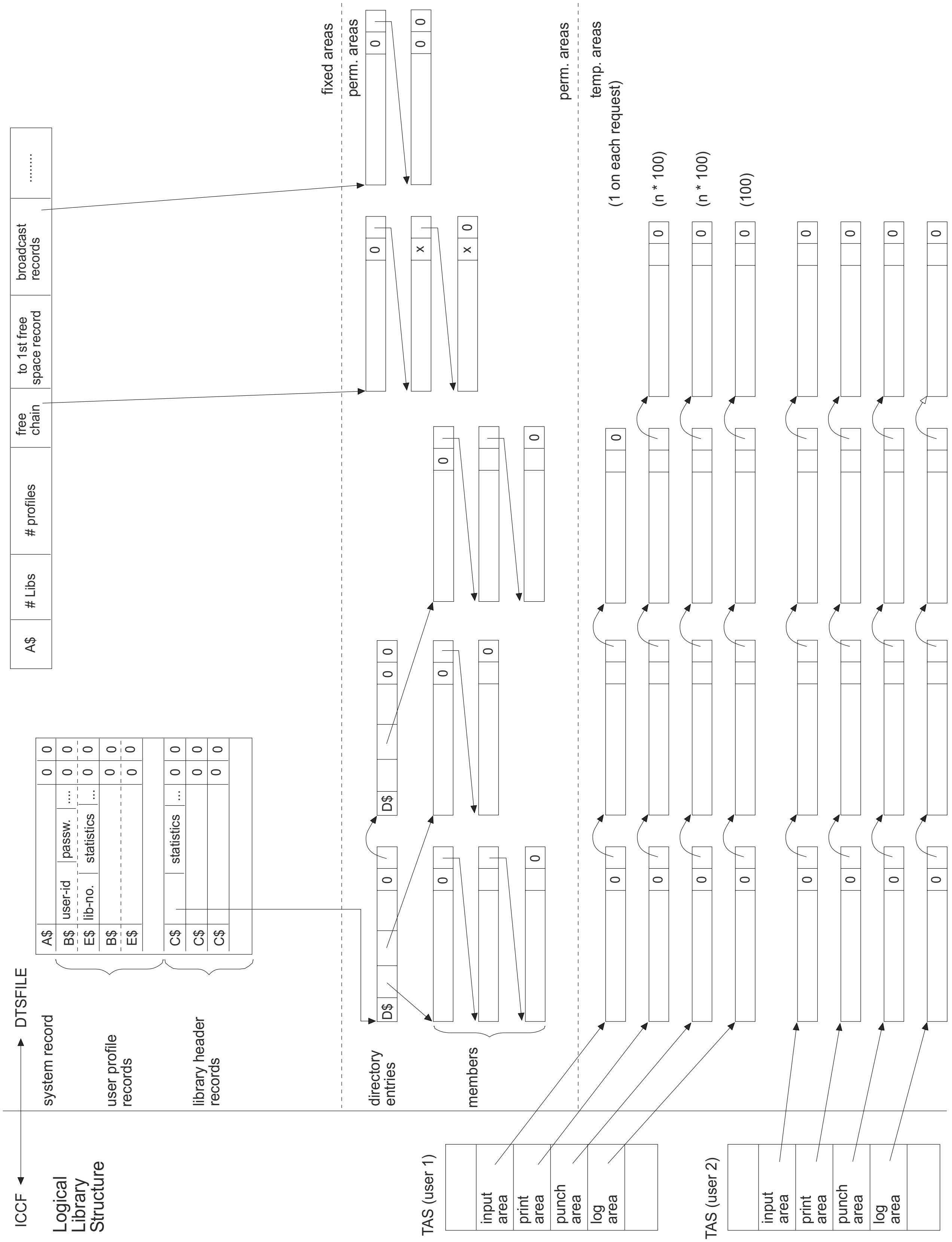


Figure 1. VSE/ICCF Library File

Each user of VSE/ICCF has access to one or more libraries, which may be owned exclusively or shared with other users. Each library has a directory which may contain any number of entries (although, for reasons of efficiency and space management, the installation may place a limit on the maximum number of entries in your library).

Each directory entry contains a name which you have given to your data. A uniquely identifiable set of data in the library is called a 'member'. Each member in the library is represented in the directory by a member name and a location in the library file. (For reasons of efficiency and space management, it is to your advantage to keep the number of members in your library as small as possible. Use the /PURGE command to remove entries from the library which are no longer needed. If you need a large number of members, use several small libraries instead of one big library.)

Each member in the library consists of one or more 80-byte records. The records may contain programs, data, object decks, procedures, macros, VSE/ICCF or VSE job streams, print or punch output of jobs that were run in interactive or in batch partitions, or any combination of these.

It is possible to logically connect members by using the /INCLUDE job entry statement. To VSE/ICCF, there is no difference between having all of the data in one member or in several members. In fact, it will be more efficient for you to update multiple small members rather than one large member. This allows you to have library members which logically exceed the limit for the number of records per library member. Each member (set of data records) which you save in your library must be identified with a name unique to that library. The name must be from 1 to 8 characters in length and the first character must be alphabetic.

Types of User Libraries

Basically there are three types of libraries in VSE/ICCF: private, public and common libraries. The main difference between these three types of libraries is the degree of protection that each offers, and this will be discussed in more detail in the following sections. The library type is recorded in the library header record. Each library is of exactly one type, and all users see this library with that type.

Public and Private Libraries

A public library is always accessible to you via the /SWITCH or /CONNECT commands. On the other hand, a private library is only accessible to you if your profile record indicates that this library is your primary library or one of your alternate ones.

Public/Private libraries should not be confused with public and private data. Refer to "Types of User Data" later in this section.

Alternate Libraries

Another way of looking at the libraries is from the way they are related to a specific terminal user. A library may be defined initially for this user as primary or as alternate library. Each user has only one primary library but he may have several alternate libraries. Data in the primary library is always accessible to the user and all data storing goes to this library. Also, data modification is permitted only for primary library data.

Data in alternate libraries is not directly accessible to a user. Such data becomes accessible when the user issues the command /SWITCH or /CONNECT. Via /SWITCH, the alternate library becomes the primary (or current) library and the original primary library is no longer accessed. Via /CONNECT, the alternate library becomes the connected (also called secondary) library. The connected library may only be used for data retrieval. A user may have only one connected library at a time.

The user profile defines, for a particular user, the primary library and the alternate libraries. The primary and alternate libraries may be private or public libraries. Public libraries may be connected by a user even if they are not in the alternate library list of that user. Access to a library does not necessarily correspond to access to data already existing in that library.

During a terminal session you are permitted to access no more than three libraries at a time: two private or public libraries or one private plus one public library, and the common library (your z/VSE comes with common library support, but this is also a tailoring option). VSE/ICCF searches these libraries in a set order and not according to the type of library concerned except for the common library which is always searched as the last one. To be able to keep track of which of your libraries you are currently searching first, and which second, you will refer to the first two libraries as your primary and secondary libraries, regardless of the type of library involved. The order of your primary and secondary libraries can, of course, be changed at any time with the /SWITCH and /CONNECT commands.

Shared versus Own Library

A library has the highest level of protection when you are the sole owner, that is, when this private library does not appear in the primary or alternate library list of another user. The user with an 'owned' private library has, of course, the same access rights as all other users, such as access to the common library, access to any public libraries and, possibly, access to libraries which are shared with other users.

A 'shared' private library, on the other hand, may be accessed by two or more users. A shared private library is one that appears in the primary or alternate library list of more than one user.

The Common Library

The common library is accessible by all users. The common library holds data which is used by several users thus avoiding duplication of data. Data in the common library is normally available only for read access, but the VSE/ICCF administrator or other authorized users may update members in the common library. Data and the corresponding directory entry exist only once in the entire library file. A library is defined as common via an operand in the VSE/ICCF tailoring macro DTSOPTNS.

If a common library exists within your VSE/ICCF environment, all directory searches will scan the common library directory if the desired member name is not found in your own primary or connected libraries. This automatic common library search may be switched off by a user if desired.

Temporary Storage Areas

As a VSE/ICCF terminal user, you not only have access to libraries but also to the following temporary library areas:

Your input area
Your punch area (also used as editor stack area)
Your print area
Your log area

These areas are part of the VSE/ICCF library file.

The Input Area

The input area may be used to receive either data entered from the terminal or data inserted from a library member.

To enter data from the terminal into the input area, the /INPUT command must be given first. This command 'opens' the input area for receiving data, it does not destroy the current contents of the input area. This is done when the first data line is entered into the input area. Once the input area is 'open', all lines, except system commands, subsequently typed in are placed into the input area. A VSE/ICCF library member or parts of it may be placed into the input area by entering the /INSERT command in input mode. When you enter /END, the input area is 'closed', the data remains in the input area. When you enter /SAVE, the input area is also closed. However, the data is saved permanently in the VSE/ICCF library, and the input area is cleared.

Once data exists in the input area, it may be added to, modified or deleted by using the VSE/ICCF editor. The editor input submode provides a means of keying in more data without having to leave edit mode.

The input area is used by VSE/ICCF as work area for the invocation of interactive partition executions. So before starting an interactive partition execution, ensure that the input area does not contain data that must be preserved.

The size of the input area is defined in the user profile.

The Punch Area

Each user has a punch area (named $$PUNCH), which contains the punched output of any jobs run in the background. The output of any program which uses the VSE symbolic units SYSPCH or SYSnnn (where nnn is defined by the installation or via the /ASSGN job entry statement as the punch programmer unit) is directed to the punch area. For example, when a VS FORTRAN (or any other) compilation is performed, the object program is placed in the punch area. From this area it may be saved in the library file or, as is normally the case, loaded into storage for execution by the LINKNGO program.

But compilers are not the only programs which can use the punch area. You may in your own programs write 80-column images to the punch area. The records thus written may be read back in a later program or stored as a member of the library.

The contents of the punch area may be physically transferred to the input area using a special form of the /INSERT command. The contents of the punch area may also be logically included as input via a similar form of the /INCLUDE job entry statement.

Punched output may also be directed to a library member rather than to the punch area, which avoids the time that would be lost in transferring the punched output to the library in a separate operation if it is to be stored.

The size of the punch area is defined in the user profile.

The Editor Stack

In edit mode the punch area is used for certain edit functions as stack area ($$STACK). Data and/or commands may be placed into the stack via the STACK editor command. Information in the stack may be retrieved with the GETFILE $$STACK editor command. If commands are saved in the stack this set of commands may be executed via a @$$STACK macro. Via an operand of the CHANGE editor command, you may request the logging (in the editor stack) of all changes made to a file. The stack may be used for saving certain editor settings. Before using the editor stack, make sure there is no data in the punch area that must be preserved. The contents of the editor stack is preserved beyond the editing session.

The Print Area

The print area, sometimes called the 'spool' area (named $$PRINT), is an intermediate storage area used to temporarily hold printed output from a background execution in order to display the output on the terminal. The default for the number of print lines which can be held in your buffer area is controlled by the VSE/ICCF administrator; a terminal user may, however, within limits, modify this value.

In command mode, the print area contents can be saved in a VSE/ICCF library member. But you can also directly route your print output to a member. Thus no time is wasted by having to transfer the print output from the print area to the library when making the print data a permanent member.

The Log Area

If you want to record details of what happens at your terminal, you may request logging. This causes your input and output at the terminal to be saved in your log area (its name is $$LOG). Then, by displaying/listing the log area, you can review the commands you had entered previously and the corresponding response messages.

The most basic level of logging stores only the command input that you have entered together with the time of entry and the mode at time of entry. Other logging options provide a more comprehensive record, including all input (commands and data) and/or all output (system responses and data).

Types of User Data

Public and Private Data

All data saved in library members is either private or public. When saving a member, the user specifies whether the saved data is private or public (the user profile specifies the default data type).

Private data may be read by all users who have access to the library with that data, but it may be modified only by the user who wrote the data initially.

Public data is accessible for reading and updating by all users who may access the library with the data. The public or private status of data may be altered via the /PROTECT command.

With alternate security, private data may be accessed, for any purpose, only by the creator of that data. Public data, however, may be read by anyone who has access to the library but modified only by the one who created the data. The concept of alternate security is discussed under "Protection of Data" on page 11.

Message Data Members

When a /SEND command is issued, the submitted messages are temporarily stored in members with reserved names. These members are automatically created for each user during logon; they may be purged using the PURGE MESSAGE command of the DTSUTIL utility.

Also, 'notification' messages from VSE/POWER are stored in these message members (the notification support of VSE/ICCF is discussed in the section "Submitting Batch Jobs to VSE" on page 17).

Members of Type Print

Members that contain print output from jobs running in batch or interactive partitions are said to be 'of type print' or are called 'print type members'. Print type members are identified by their name, which ends with '.P'. No other members should therefore have a name that ends with '.P'. Print type members contain control characters which allow them to be displayed (/DISPLAY, /LIST) and printed (with the PRINT macro) in print format.

Generation Member Group

With generation member groups it is possible to create versions of VSE/ICCF library members, that is, when a member is saved or replaced it becomes the current version and the previous current version becomes the old one. Up to 10 different versions of a member may be kept. With the /GROUP command the maximum number of versions is defined. If the maximum number of versions has been saved within a group and a new version is to be saved the oldest version will be purged.

Common Data

Common data is data which appears only once in the entire library file, but a directory entry for this data exists in each library which is defined as a library that may hold common data directory entries. Common data is accessible to all users for reading; writing is allowed only for authorized users.

For information on how to define and create common data, refer to the DTSUTIL functions ADD/ALTER LIBRARY and SHARE MEMBER.

Dates in VSE/ICCF Library Records

The VSE/ICCF library (DTSFILE) contains 4 types of directory records. All of them contain date entries. The system record (A$), the user profile records (B$), and the library header records (C$) include additional century information in decimal format. The directory entries for members (D$) have a long history of changing date formats that VSE/ICCF must still be able to interpret. The year is stored as a one-byte binary number (relative to 1900).

Field RTDDATE (D$ Record)

The length of this field is 5 bytes and it contains the date in the format *ymmdd*.

The following list shows the hexadecimal values for the year representation (Y). GE means Greater or Equal, and LE means Lower or Equal.

```
                Y =  00  invalid
          01 GE Y LE EF  1901 to 2139
          F0 GE Y LE F4  1980 to 1984
          F5 GE Y LE F9  1975 to 1979
          FA GE Y LE FF  invalid
```

X'F5' (1975) represents the introduction of VSE/ICCF.

The User Profile

Each user of the system has a unique, four-character user identification. Associated with this user identification are two 80-character data records referred to as the user profile record. Some typical fields in your user profile record are:

- Your user identification code.
- The 'logon' password (in scrambled format)
- The name of your logon procedure or macro if one exists.
- The identification of your primary library.
- The identification of any alternate libraries to which you may have access.
- The maximum allowable sizes of your punch and print areas.
- Your time limits for any given execution.
- Accounting information such as number of logon requests, number of commands entered, number of execution (background) requests and execution time.
- Library usage information.
- Your security level.
- Your privilege class (for example, 'VSE/ICCF administrator').
- Whether prompting by line number is the default.
- Whether you may enter jobs into background execution or whether you are limited to submitting to 'batch' for compilation and execution.
- Whether your data is entered into your library as public or private data if not explicitly stated.

The user profile record, thus, contains the information necessary to inform the system about your requirements and security level.

Protection of Data

VSE/ICCF provides facilities which protect libraries, library members, files and phases against unauthorized access from interactive partitions.

The following is an overview of these facilities:

- User identification code – Each user of VSE/ICCF has a unique user identification (ID), which must be specified when logging on to the system. This ID is also used to identify the owner of private members in a library which is shared by other users.
- Logon Password – A three to six character password must be specified for logon. This password should be changed from time to time by the VSE/ICCF administrator to ensure continued protection. The password should be unique for each user. Certain users may be entitled to change their passwords themselves via the /PASSWRD command.

  z/VSE supports two security components. The basic security manager (BSM) and the external security manager (ESM). These components maintain userids and passwords. VSE/ICCF still maintains userid and password in the DTSFILE. This is only for compatibility reasons. Following rules are in place for an user's sign-on to VSE/ICCF:

  - Logon from the Interactive Interface is always allowed.
  - Native Logon to VSE/ICCF requires either the password from the DTSFILE or the central password from BSM or ESM.

  VSE/ICCF uses every chance to keep the password in the DTSFILE in sync with the central password.

- Shared versus Own Library – The highest level of data protection is attained when you own a library exclusively. Access to an owned library, for any purpose whatever, is only possible via the owner's ID. A shared library, on the other hand, is accessible to a number of users, as identified by their user IDs. Data in a shared library can be modified by other users of the library, provided the data has been entered as public. To protect data in a shared library, you must enter it as private, or protect it with a password.
- Primary and connected libraries – Members may be saved or updated only in the primary library, but directory searches access the connected library (and the common library) when looking up members for retrieval.
- Public versus Private Data – All data that you save in your library is either public or private, regardless of whether you specified this option in the /SAVE command. The default (PUBLIC or PRIVATE) is indicated in your user's profile record. Public data may be accessed, for any purpose, by any user who shares the library (unless the 'alternate security' option is in effect). Private data may be read by any user who shares the library (unless the 'alternate security' option is in effect), but it can only be modified by the user who entered it.
- Normal Security versus Alternate Security – With normal (default) security, public data may be read or updated by anyone who shares the library. Private data may be read by any user of the library but only updated by the user who entered the data.

  Under the alternate security approach, public data may be read by any user but updated only by the user who entered the data. Private data, on the other hand may not be accessed for any purpose except by the user who entered the data.
- Common Data – Common data is public in the sense that all users have read access to it. However, you may not normally update common data; only the VSE/ICCF administrator may do this, although you may insert common data into your input area, update it, and save it in your library under a different name.
- Member Passwords – Any user may save members in the primary library with a four-character password. The password must not be PRIV or PUBL and must begin with an alphabetic character. Any subsequent reference to the member name must include the password as a suffix. For example, /LIST FORTPROG PASS would list the member FORTPROG which was protected with the password PASS.

  If a password protected member is flagged as public, any user who has access to the library and who knows the password may modify the data. The password of a member in the library may be added, changed or deleted using the /PROTECT command. The use of a password reserves all forms of access (read and modify) for the user who entered the member.
- Protection for User Files and Programs – Tables are provided in VSE/ICCF in which restricted files and programs may be defined and to which only authorized users have access.

  The System Program table and the Load Protection table protect phases against unauthorized access. User files are protected via the System File table.

  Phases and user files can also be protected through a non-VSE/ICCF facility: the Access Control function of the VSE System (which must be activated at IPL time). The *access control table* (DTSECTAB) of the VSE System is used to control the access to user files and to VSE libraries, sublibraries, and members in a sublibrary. The optional program *VSE/Access Control-Logging and Reporting* extends the VSE Access Control function. For more information on these protection facilities refer to:

  – z/VSE Administration
  – VSE/Access Control-Logging and Reporting: Program Reference and Operations Guide

The Editors

VSE/ICCF includes two different editors:

- The *full screen* editor, which can be used only with the IBM 3270 Information Display System, or with terminals that act like an IBM 3270.
- The *context* editor, which can be used with any input/output terminal.

All editor commands have an immediate effect on the data that is being edited. When you request a change to this data, VSE/ICCF makes the change to the actual file.

The Full Screen Editor

This editor was designed to work with the IBM 3270 Information Display System; it provides you with all the features that these terminals offer, such as user defined program function key options and minimum data transmission (changed data only). The full-screen editor provides support for:

- Split screen control, allowing multiple displays.
- Concurrent editing of several files in a single session.
- Concurrent editing of different areas within a given file.
- Editing of the print and punch areas.
- Creation of new library members while in edit mode.
- Formatting of the displayed data.
- Display or modification of data in HEX or EBCDIC format.
- Moving and copying data on the screen or between screens.

With the split screen facility you are able to display and edit up to eight files, or different areas of the same file, on the screen at one time. You can move or copy data within the same file, or between files, either on the present screen or onto subsequent screens. If extra room is needed to view a particular file, the displays for the other files can be switched off the screen and restored when they are needed. Any number of files can be edited during a given editing session, and new library members can be created without having to leave the editor environment.

The Context Editor

The context editor, a general purpose text manipulation program, allows you to create and modify card image files from the terminal. A file may be a member of your library, or you may edit one from your input area. The context editor must be used if:

- You work with an IBM 274x terminal.
- You write procedures and macros that perform editing functions.

The context editor consists of commands and macros that locate and modify strings of information within a file. You can use these commands and macros to move a logical pointer from one record to another and to perform other control functions such as setting logical tabs or the string scanning zone, replacing or inserting data lines and editing files by line number.

If you are using a display screen terminal, your location within the member being edited is indicated by a second scale line across the screen. The current line is the line directly below the scale line. You select the number of lines (1-20) you wish to appear before the current line and the number of lines (1-20) you wish to appear after the current line. These values are selectable during editing. All lines below the scale line appear in heightened intensity. You may turn full screen verification on or off. ON is assumed for a local IBM 3270 Display System.

Editing Facilities

The following paragraphs introduce you to some of the editing facilities of VSE/ICCF:

- The editor stack facility – The editor stack is a scratch file work area which may be used for several different purposes during editing. For example, you may execute commands from the stacks. If you wished to repetitively issue four commands at various places within your member, you should place these commands in the stack. Then whenever you wish to execute these commands, all you have to do is type in @$$STACK. The commands in the stack contain variables which are replaced by values from operands when the stack is invoked.

  The stack may be used also for temporarily storing lines from the member being edited for insertion at some other point in the file. When issuing global change commands, you can specify that the editor is to place in the stack all lines on which a change will be made. Thus, you may review the lines actually changed during a global change.

- Changing data in the input area – Entering the CHANGE command with no operand from an IBM 3270 Display System causes the current line to be placed in the input area of the display. It may then be altered and replaced in the member. You may use a column suffix to indicate where the cursor should be placed. For example, 'CC36' causes the current line to be placed into the input area and the cursor positioned at column 36.
- Text manipulation – Certain commands are provided within the editor for carrying out text manipulation functions such as centering data within the zone (CENTER), shifting data within the zone (SHIFT), right or left justification within the zone (JUSTIFY), right and left justification within the zone (ALIGN) and splitting a given line into two lines so new text may be inserted (SPLIT).
- Flagging changes – Library members may be established such that all editor functions which change data within the member cause the change to be noted in the data lines themselves. For example, you may desire that changes be noted in columns 73-80. If so, the user causing the change, the data of the change and the type of the change are placed in columns 73-80.
- Inserting data from other members – The GETFILE command allows you to insert all or a portion of a given library member at any given point within the member being edited. GETFILE may also be used to insert data from the stack or to insert data from another part of the current member.
- Line-number editing – If you use the editor in line mode, you can do line-number editing as well as context editing. Finding a given line in line-number editing requires no more than typing the line number and pressing ENTER. To replace a current line or to add a new line, just type the line number followed by the data and press ENTER. The editor automatically positions your file to the specified line and makes the desired addition or replacement. Line-mode editing and normal context editing may be used concurrently on the same file.
- Index for large members – For editing large members, you may create an index for the member. The index enables you to move the line pointer quickly. For more information on indexed editing, refer to the VSE/ICCF User's Guide. When line-number editing is used with indexing, finding a line-number position in the member is also done using the index.
- Moving and copying of data from one part of a member to another is done by way of @MOVE and @COPY editor macros. For example, '@COPY 10 LOC /IDEA/' causes the ten lines at the current position to be copied behind the first line where the string 'IDEA' occurred.
- Locate functions – Aside from the normal LOCATE forward functions, the LOCATE command offers an upward locate function (move pointer towards top of file) and a 'locate not' function to stop on the first not equal within the zone.
- Logical edit procedures – The &&IF statement within the procedure processor has the ability to interrogate the current editor line for a given data string. Thus, special-purpose edit procedures which require complex logical condition evaluation may be prepared. For example, change the string /XXX/ to the string /YYY/ in all lines of the file which contain the string /AAA/ but not the string /BBB/.
- Scrolling – A member being edited may be scrolled on the display screen while the logical pointer is being advanced. Scrolling may be carried out by pressing the ENTER key or automatically by a timer interval value. When scrolling based upon a timer interval, the PA2 key may be depressed to terminate scrolling. The current line pointer will be set to the last line displayed. If you are using a display screen you may, for example, display a library member and specify a time interval; at the expiration of this interval, a new display is to appear at the terminal.
- Physical Screen and Logical Screens – The physical screen of an IBM 3270 Display System may be divided into from two to eight logical screens. Within each logical screen, a single file may be edited. Thus, it is possible to edit up to eight different files concurrently on the same physical screen of your IBM 3270

  There are many reasons why a user might wish to view, or view and edit, two or more files concurrently:

  - When copying or moving data from one file to another, a user may find it helpful to see both files at the same time without having to leave the edit environment.

  - When correcting compilation errors in a program previously compiled in an interactive partition, the user may ENTER $$PRINT in order to locate his errors. He may scroll through his errors at the same time he is editing the errors out of his program.
  - When making changes to a program, it may be helpful to be able to see an associated member. For example, when changing a program, a user may wish to see one or more COPY books or DSECTs associated with the program.
  - While a file is being edited, the user may decide that he wishes to build a new macro or modify an old macro to aid him in his edit session. To do this, he merely ENTERs the macro, types in the macro statements, SAVEs the macro in the library and reenters the file in which the new or modified macro is to be used. He may now use this macro in his edit session.
- Logical Screens and Format Areas – Any given logical screen may be divided into from 1 to 8 format areas. With a format area, a portion of a file may be edited. Thus, it is possible to view and to edit up to 8 completely independent areas within the same file and have all on the screen at the same time. This facility is useful when you want to view one area of a file while making changes to another area or want to make matching changes to multiple areas or when moving or copying data from one portion of a file to another.
- Editing a Non-Existent File – When you edit a non-existent file for the first time, the file contains no records. Records may be added to the file by any number of means. Among them are the INPUT, LADD and GETFILE commands. At this point, the file is not a library member and it is not the input area. It is simply a new file built by the Full Screen Editor. If, at some later point in the edit session, the user decides he wishes to retain the newly created file in the library, he may issue the SAVE or FILE command. Several new library members can be built in this manner without leaving the editor environment.
- The Selective Viewing Facility and the VIEW Command – If a user wishes to change the way the data from the records in the file is displayed on each line, he may use the VIEW command to select and/or rearrange the columns which are viewed.

  The command allows the user to:

  - View only certain columns of the 80 character record on the screen. For example, to view only columns 12 through 80.
  - Rearrange the order in which data is displayed. For example, to view columns 41 to 80 followed by columns 1 to 40.
  - View certain fields in hexadecimal format rather than character format.
  - Place blank filler fields between certain fields displayed on the screen.
- In full screen edit mode, program function keys may be equated to data lines, to commands, or to multiple commands or data lines.
- Cursor positioning functions may be equated to program function keys to allow easy positioning of the cursor to any location on the screen.
- Data lines may easily be moved or copied from one part of a screen to another part of the screen which may be within the same file or a different file.
- Data lines may be collected from one or more screens for insertion into one or more subsequent screens.

Dynamic Space Allocation

To save you from having to pre-allocate disk work file space at VSE/ICCF startup time, VSE/ICCF offers dynamic space allocation. It allows you to allocate this space (from a pool of disk space) when you are scheduling the job for execution in an interactive partition. Dynamically allocated files may be specified as permanent or temporary, that is, they can be retained from day to day or they may be accessible only during execution of the job or job step that allocates them. The retention of these files will be subject to central installation procedures; that is, if all dynamic space areas are 'cold' started each day, no user files may be retained.

Job Streams

Job streams may be built in the input area and executed with the /RUN command, or saved as members of the library and executed with the /EXEC command. A job stream consists of job entry statements (such as /LOAD, /FILE, /OPTION, /UPSI and /ASSGN) and may also consist of source programs and/or data. If the source programs and data to be processed within a job stream reside as separate members of the library, they may be logically included in the job stream using the /INCLUDE statement.

A job stream may consist of a single program execution (single step job) or multiple program executions (multistep job). Each step in a job stream begins with the /LOAD statement unless an execution of the LINKNGO utility is implicitly requested in which case a /LOAD statement is not required.

You may define existing VSE files (SAM, DAM, VSAM) in your VSE/ICCF job streams via the /FILE statement, thus eliminating the need to have all DLBL/EXTENT cards present in the VSE/ICCF startup deck. In addition to being able to define VSE files via the /FILE statement, you may also define dynamically allocated files, and assign unit record devices to various 'SYS' units rather than having to use the system defaults. You may also cause a job stream to conversationally prompt the terminal for data at execution time. If errors are detected in job entry statements, the terminal is prompted for reentry of the statement in error. VSE/ICCF will allow programs running in interactive partitions to set the VSE operator communication exit. The exit is activated by the /ATTEN command or by the 2741 ATTN key or a 3270 PA key.

The INCLUDE Facility

When you build a job stream, the programs to be compiled or the data itself, or both are usually part of this stream. However, it is often more convenient to have programs and/or data broken down into small, separate modules. To group several members together into one member, use the /INCLUDE statement.

Normally you might enter a FORTRAN compile and execute with the following job entry statements and data card images.

```
/LOAD VFORTRAN
  (FORTRAN source program)
/DATA
  (Input data card images)
```

However, if the data was in a separate member of the library named FORTDATA, the job entry stream might look as follows:

```
/LOAD VFORTRAN
  (FORTRAN source program)
/DATA
/INCLUDE FORTDATA
```

Or, the program itself may be in yet another member named FORTPROG in which case the job entry statements would look like this:

```
/LOAD VFORTRAN
/INCLUDE FORTPROG
/DATA
/INCLUDE FORTDATA
```

Now, if the FORTRAN program was large and was stored in three library members named FORTPR1, FORTPR2 and FORTPR3, the following job entry stream would result:

```
/LOAD VFORTRAN
/INCLUDE FORTPR1
/INCLUDE FORTPR2
/INCLUDE FORTPR3
/DATA
/INCLUDE FORTDATA
```

/INCLUDE requests may be nested eight levels deep. That is, it is permissible to have /INCLUDE statements within library members which are themselves the object of an /INCLUDE.

Procedures and Macros

Procedures (sometimes referred to as command lists) and macros are VSE/ICCF library members which contain a sequence of statements that together carry out frequently used functions such as compilation, loading and execution of programs, storing object decks and sorting library members. The statements in a procedure or macro may be VSE/ICCF system commands, context editor commands, job entry statements, procedure processor orders or macro orders, or data.

The purpose of such procedures and macros is to save you from having to reenter the same series of commands and statements each time you want a particular function. Once the procedures or macros have been created and stored, they can later be invoked easily. A procedure is invoked by entering the member name. A macro is invoked by entering

- The member name, if you are in command mode;
- The member name prefixed by an @, if you are in edit mode.

The main difference between procedures and macros is that a procedure may contain logical statements which determine the flow and execution of the commands, while a macro has less logic capability. Another difference is that macros are executed in the foreground like normal commands and may be invoked while in edit mode, whereas procedures are executed in the background and can only be invoked in command mode. Because procedures require the procedure processor program to be executed in an interactive partition, they are slower than macros.

Parameter substitution is possible for both procedures and macros, which allows you to supply them with variable data, thus increasing their flexibility.

A single macro may be created that contains groups of system or editor commands and executed with one terminal operation, thus greatly extending editor flexibility. For example, a macro may be set up to place your terminal in context edit mode, to set tabs, indexing, verification and other defaults and return to the terminal to continue the edit process. Temporary macros may also be created during editing and executed during the same edit session. The commands to be edited are placed in an area called the 'stack' area and executed from there. Commands in the stack may be executed repetitively any number of times.

IBM-supplied macros and procedures are made available with VSE/ICCF, but facilities are also provided in the system that enable you to write your own procedures and macros. A typical IBM-supplied procedure is ASSEMBLE, which assembles an assembler-language program and produces an object module. An example of an IBM-supplied macro is LIBRL, which may be used to display a member of a VSE sublibrary. For more information about procedures and macros see "Writing Your Own Procedures and Macros" in the VSE/ICCF User's Guide.

Submitting Batch Jobs to VSE

Besides enabling you to run batch jobs in the VSE/ICCF environment, VSE/ICCF also allows you to submit VSE/ICCF and VSE jobs to VSE/POWER controlled partitions for processing, provided VSE/ICCF is running in a system that includes VSE/POWER.

Using the SUBMIT facility, you can enter jobs into the VSE/POWER reader queues just as you would present normal VSE jobs for processing via VSE/POWER. You can display the status of these jobs and also the entries in the various VSE/POWER queues at any time during execution.

Print output from such a job may be written directly to the main system printer, or it may be displayed at your terminal before being routed to the system printer, to a VSE/POWER RJE printer, or to a printer associated with your terminal. With punch output, you have similar options. Print output and punch output can also be retrieved from the corresponding VSE/POWER queue with the GETL and GETP procedure and saved as a member of a VSE/ICCF library. Similarly, jobs that have been submitted to VSE/POWER can be retrieved from the VSE/POWER reader queue with the GETR procedure.

The job stream can be set up such that the submitter of the job is notified after the job has finished processing. To request this service, the submitter specifies NTFY=YES in the * $$ JOB statement of VSE/POWER. The notification either appears automatically at the submitter's screen (if automatic message display mode is in effect) or the submitter must explicitly ask for it by issuing the /MSG command.

If your VSE/ICCF runs on a processor that is part of a network of processors, you can route the submitted job to a processor different from the one where the submission takes place. Likewise, the output of the job can be routed to a remote destination. This service is available through the PNET generation option of VSE/POWER. Also, the VSE/POWER job entry control language (JECL) must be set up accordingly (see the model JECL statements in the IBM-supplied "Submit to Batch (DTSSUBMT) Program" on page 83).

VSE/POWER Queue Display Facilities

- Display a job's printed output from the VSE/POWER print queue on your terminal.
- Display VSE/POWER queues on your terminal.
- Retrieve data from the VSE/POWER reader, punch, or print queue and place it in the VSE/ICCF library file.
- Locate a specified character string in the VSE/POWER print output and display the lines following.
- Display the execution status of submitted jobs.
- Route printed/punched output of a submitted job to the main system print/punch unit or to a remote VSE/POWER RJE terminal print/punch unit.
- Delete a job from the VSE/POWER reader queue.
- Delete printed/punched output from the VSE/POWER print/punch queue.
- Authorized users may enter, at their terminal, any VSE/POWER command (except VSE/POWER task management commands such as PEND, PFLUSH, PSTART).
- Include data from diskette in the input stream.

Note: If your location uses VSE/POWER job accounting, ensure that VSE/POWER's account file is large enough. Every display of a job entry of the VSE/POWER print queue causes an account record to be written into the account file.

Languages Supported

VSE/ICCF attempts to create a background processing environment for some 'batch' (that is card-in, printout) type compilers to operate without modification.

Therefore, you may write and execute programs in the following languages: HLASM, FORTRAN and RPG*.

Specifically, the language compilers which can be used under z/VSE are VS FORTRAN, High Level Assembler for VSE and DOS/VS RPG II .

Support for COBOL and PLI has been removed, since ICCF does not support Language environment (VSE/LE) in its interactive partitions.

Debugging Facilities

VSE/ICCF provides several dump programs:

```
DTSCDUMP
DTSFDUMP
DTSSNAP
DTSPGMCK
```

DTSCDUMP enables you to investigate failures in a program that runs in an interactive partition. The DUMP option of the /OPTION command causes this dump program to be invoked automatically if a program terminates abnormally. The program displays various items of information about the termination, such as the PSW, the general registers and data fields associated with the terminating instruction, etc. You may then enter other commands to display various storage areas within your program in order to locate the source of the problem. If desired, a hardcopy dump of an interactive partition can be obtained for later problem determination.

DTSFDUMP helps you to investigate error conditions within VSE/ICCF itself. It formats VSE/ICCF internal tables and sends formatted output to SYSLST. DTSFDUMP runs either in a VSE partition or in an interactive partition of VSE/ICCF. You can obtain the DTSFDUMP functions also by issuing the VSE/ICCF operator command /FDUMP.

"DTSSNAP" and "DTSPGMCK" are subroutines that you can call in your program to display certain portions of main storage and the general registers. Both subroutines are described in the VSE/ICCF User's Guide.

IBM-Supplied Procedures and Macros (and Samples)

z/VSE contains VSE/ICCF procedures, macros and some sample programs (plus associated members). Their names are listed in Table 1 on page 19. Column 2 of this figure indicates where the members reside:

C
Denotes the common library, which is library 2.

A
Denotes the VSE/ICCF administrator's library, which is library 1.

You may use the supplied procedures and macros as examples for writing procedures and macros of your own. For performance reasons, these macros and procedures have been kept short; they neither check nor handle all situations that could occur.

Table 1. IBM-Supplied Procedures, Macros and Samples

Name	Lib	Type	Function
A$HELP	C	member	Help information.
A$MAIL	C	member	Example for mail.
ASSEMBLE	C	procedure	Assemble an assembler-language program and produce an object module.
COPY	C	macro	Copy 1 to 99 lines from one portion of the file to another (edit mode).
COPYFILE	C	macro	Copy a VSE/ICCF library member to your primary library under a different name.
COPYMEM	C	procedure	Copy a library member from one library to another library with (an optional) new name.
CPYLIB	C	procedure	Copy a library member from one library to another library.
DTSLG00	C	macro	Example of a logon procedure for a typewriter terminal.
DTSLG4R	C	member	Example of a logon procedure for a typewriter terminal.
DTSLG40	C	macro	Example of a logon procedure for a typewriter terminal.
DTSLG7A	C	member	Example of a logon procedure for a screen-type terminal.
DTSLG70	C	macro	Example of a logon procedure for a screen-type terminal.
ED	C	macro	Directly invoke the full-screen editor.
EDPRT	C	macro	Invoke the full-screen editor for the print area.
EDPUN	C	macro	Invoke the full-screen editor for the punch area.

Table 1. IBM-Supplied Procedures, Macros and Samples (continued)

Name	Lib	Type	Function
FORTPROG	A	procedure	Sample VS FORTRAN compile and execute (included only in the VSE/ICCF administrator's library).
FORTRAN	C	procedure	Compile a FORTRAN program with the VS FORTRAN compiler and produce an object module.
FSEDPF	C	macro	Set PF keys for use in a full-screen editor environment.
GETL	C	procedure	Move job output from the VSE/POWER list queue to a VSE/ICCF library.
GETP	C	procedure	Move job output from the VSE/POWER punch queue to a VSE/ICCF library.
GETR	C	procedure	Retrieve a job from the VSE/POWER reader queue and store this job in a VSE/ICCF library.
HC	C	macro	Switch to hardcopy mode, execute a specified command and return to command mode.
HELP	C	macro	Invoke help panels.
HELP$LIS	C	macro	Used by HELP macro.
HELP$LST	C	macro	Used by HELP macro.
LIBRC	C	macro	Catalog a member of a VSE/ICCF library as a member of a VSE library.
LIBRL	C	macro	List a member of a VSE library.
LIBRP	C	macro	Punch a member of a VSE library.
LOAD	C	procedure	Load and execute an object module.
MOVE	C	macro	Move one portion of the file to another, up to 99 lines (edit mode only).
MVLIB	C	procedure	Move a library member from one library to another.
PRINT	C	macro	List a library member on the hardcopy printer.
RELIST	C	macro	Print on a central-site printer the print area contents or a member containing print output.
RPGIAUTO	C	procedure	Call the RPG II autoreport service.
RPGII	C	procedure	Compile an RPG II program and produce an object module.
RPGIXLTR	C	procedure	Call the RPG II DL/I translator.
RSEF	C	procedure	Invoke the RPG II Source Entry Facility.

Table 1. IBM-Supplied Procedures, Macros and Samples (continued)

Name	Lib	Type	Function
SAMPASMB	A	procedure	Sample assembly and execute (contained only in the VSE/ICCF administrator's library).
SAMPFORT	A	procedure	Sample VS FORTRAN compile and execute (contained only in VSE/ICCF administrator's library).
SCHD1ASM	A	member	}
SCHD2ASM	A	member	}
SCHD1COB	A	member	}
SCHD2COB	A	member	}
SCHD1PLI	A	member	}
SCHD2PLI	A	member	}
SCRATCH	C	procedure	Scratch a user-permanent (DISP=KEEP) file from a dynamic space area.
SDSERV	C	procedure	Produce a sorted VSE/ICCF library directory display.
SORT	C	procedure	Sort a library member.
STORE	C	macro	Store the contents of the punch area under a specified name.
SUBMIT	C	procedure	Submit a library member for execution in another VSE partition.
WORKFILE	C	member	Needed to run sample programs.
WORKFIL2	C	member	Needed to run sample programs.
WORKFIL3	C	member	Needed to run sample programs.
ZZZ$...	C	members	Help panels used by the HELP macro.

Chapter 2. Tailoring VSE/ICCF

This section describes a number of tailoring options and other facilities within VSE/ICCF that you can activate to best utilize the program.

Note: VSE/ICCF is operational once you have z/VSE installed, and it might serve your needs without any further modifications.

For the remaining chapters take notice of the way that z/VSE has packaged VSE/ICCF:

1. VSE/ICCF as a program is stored in the VSE system sublibrary IJSYSRS.SYSLIB.
2. z/VSE uses some of the VSE/ICCF libraries for special purposes of its own. Library 59 has several skeletons some of which were created to help the VSE/ICCF administrator.
3. The VSE/ICCF library file resides on volume SYSWK1. Its filename is DTSFILE, its file ID is ICCF.LIBRARY.

VSE/ICCF Tailoring Options (DTSOPTNS Macro)

This section describes the tailoring options of VSE/ICCF. If you do not intend to change the preset options you may skip this text.

This section tells you which tailoring options are available to you. You change the tailoring options by running a tailoring job. The next section shows you how to set up the tailoring job.

You specify your tailoring options as operands of the DTSOPTNS macro. Table 2 on page 23 lists all the options that may be specified in the DTSOPTNS macro. It indicates the possible operands, with the defaults underlined. The defaults are also the preset values of the distributed system. The preset options of the distributed system are available in the system library as

source book DTSGENER.A
phase DTSIGEN.PHASE

Table 2. Summary of DTSOPTNS Tailoring Options

Operand	Specification in DTSGENER.A
ALTSEC=[NO|YES]	NO
ATN2741=[NO|YES]	YES
CANKEY=[PA1|PA2|PA3]	PA2
CISIZE=[2048|n]	2048
COMLIB=[2|n|NO]	2
CRJE=[NO|YES|(YES,Q,A,D,A)|(YES,x,y,z,r)]	(YES,Q,A,D,A)
DISPKEY=[PA1|PA2|PA3]	PA3
DYNSPC=[NO|YES]	NO
EDFLAG=[73|n]	73
EDEND=[72|80]	72
FILEVER=[NO|YES]	NO
HCLINE=[132|n|NO]	132
INTCOMP=[NO|YES]	YES
INTRVAL=[1|2|3]	1

Table 2. Summary of DTSOPTNS Tailoring Options (continued)

Operand	Specification in DTSGENER.A
KATAKAN=[NO|YES]	NO
LOADPRT=[NO|YES]	YES
NBUFS=[20|n]	20
NRECS=[22|n]	22
NUSRS=[30|n]	30
NPARTS=[5|n]	5
NTASKS=[4|n]	4
PGMRINP=[5|n]	5
PGMRLST=[6|n]	6
PGMRPCH=[7|n]	7
PGMRPIN=[8|n]	8
PGMRLOG=[9|n]	9
PSIZE=[256|n]	256
PARTN=(n,xxx,y,zzzz...)	See Note below
PARTX=(n,xxx,y,zzzz...)	
RDR=[FFC|xxx]	FFC
RDR2=[FFA|xxx]	FFA
PCH=[FFD|xxx]	FFD
PRT=[FFE|xxx]	FFE
SPOOL=[250|n]	250
TCUPSI=00000000	
TCTOFS=[8|n]	8
TIOA40=[600|n]	600
TIOA00=[600|n]	600

Note: The specification used in DTSGENER.A is as follows (read this as a contiguous string):

```
   (1,1024,4,I,
    2,384,4,A,
    3,384,4,A,
    4,512,4,BA,
    5,512,4,BA)
```

Description of VSE/ICCF Tailoring (DTSOPTNS) Options

ALTSEC=[NO|YES]

Specify YES if you want the 'alternate security' or NO for normal security. NO is the default. Under normal security, private data may be read by any user but only updated by the originator. Under the 'alternate security' option, private data may not be accessed for any reason whatsoever except by the originator, while public data may be updated by the originator only but may be read by any user of the library.

ATN2741=[NO|YES]

Specify YES or allow the default if IBM 2741s or compatible IBM 3767s are supported and attention key support is desired. If NO is specified, the attention key on a read from the IBM 2741 will not be distinguishable from the carriage return key.

CANKEY=[PA1|PA2|PA3]

Specify the 3270 PA key (PA1, PA2 or PA3) which is to be treated as the cancel key for canceling interactive partition executions and for terminating edit, list, and input modes. The default is PA2 which should be used where possible because VSE/ICCF publications refer to the PA2 key as performing the cancel functions. Never specify this key as the same one that is used as a CICS Transaction Server print request key.

CISIZE=[2048|n]

The control interval size is required to process the VSE/ICCF library when it resides on an FBA device. The value you specify for n must be a multiple of 512, the size of a block on an FBA disk; it must be a multiple of 2K if greater than 8K. The maximum value is 32,768. The CISIZE value is placed in the format-1 label area of the VTOC on the library disk pack by the VSE OPEN macro during the formatting of the VSE/ICCF library file. It can be altered only by reformatting the VSE/ICCF library file because VSE/ICCF library record calculations and buffer size allocations are based on this value. Also a library created on a system with a certain CISIZE specification cannot be used on another system with a different CISIZE specification. The number of buffers allocated via GETVIS is determined by the NBUFS= option specification (see below).

Keep in mind that a large CISIZE specification in combination with the NBUFS value and the related buffer control block areas necessitates a correspondingly large I/O area in storage. The number of VSE/ICCF logical records per control interval can be determined by the following formula:

```
Records per CI=(CISIZE-10)/88 where 88 is the VSE/ICCF
logical record size and the variable is the CISIZE.
```

For example, using the 2048 default CISIZE:

```
(2048-10)/88=23 records per CI
```

In the above example, 23 logical records will be read or written per physical I/O for the VSE/ICCF library on an FBA device.

COMLIB=[2|n|NO]

If NO is specified for this option, your VSE/ICCF will have no common library support. To specify common library support, enter the number of the library which is to be treated as the common library. A common library is useful because it is available to all users for read access. Macros and procedures supplied with VSE/ICCF usually reside in a common library. If a common library is specified, all directory searches will scan this library if the requested item is not found in the user's primary or connected libraries.

CRJE=[NO|YES|(YES,Q,A,D,A)|(YES,x,y,z,r)]

This operand controls whether foreground commands (for example, /LISTP or /DQ) may be used:

1. To display output from VSE/POWER at the terminal,
2. To display the status of VSE/POWER queues, or
3. To control VSE/POWER queues.

The other facilities, that is: SUBMIT, GETL, GETP, and GETR are procedures and may, therefore, be used even if CRJE=NO has been specified. The same is true for the RELIST macro.

The specification (YES,x,y,z,r) allows the terminal user to issue foreground commands in order to communicate with VSE/POWER.

x

This is the print queue class in which printed data is expected by the command processors and by DTSGETQ for display at the terminal. The default is 'Q'. This should be a class for which no VSE/POWER list task has been started.

y

This is the print queue class in which the printed data is placed when the user has finished displaying the data at the terminal and wants to have it sent to the central system printer. Default is class 'A' which in most installations is the default print class.

z

This is the disposition which is set for the printed data once it has been routed to the system printer. The default is 'D', which causes the data to be automatically purged from the queue when printing is complete. Specify 'K' if you want the printed output to be kept in the queue until purged by the operator or by the terminal user. Specify 'H' to prevent VSE/POWER from printing the output until instructed to do so by the operator or the terminal user.

r

This is the class in which reader queue entries are expected to be if no class is specified. The default class is 'A'.

If CRJE=YES is specified, CRJE= (YES,Q,A,D,A) is assumed (see also section "Submit to Batch (DTSSUBMT) Program" on page 83 to establish POWER® model JECL).

The GETL and GETP procedures use as default output class for printed/punched data the value which has been specified for 'x'. If CRJE=NO is specified, GETL and GETP assume Q as output class.

DISPKEY=[PA1|PA2|PA3]

VSE/ICCF assumes the availability of PA keys. The PA3 key is the default display/attention key, but these assumptions can be changed by the 'CANKEY' and 'DISPKEY' operands of the DTSOPTNS macro.

If any of your IBM 3270 Display Systems have only PA1 and PA2 keys and one of these two keys is defined as the print request key in CICS Transaction Server (PRINT operand in the DFHSIT macro), then this key cannot be used for any other purpose. You must give up either the VSE/ICCF cancel key function or the display/attention key function. Of the two functions, the display/attention key function is probably less useful and should be given up. You do this by specifying the 'DISPKEY' operand equal to the PA key assigned to the CICS Transaction Server PRINT operand of the DFHSIT macro. If some of your IBM 3270s have PA3 keys, you can assign this key to the 'DISPKEY' function.

Terminals with a PA3 key do have a display/attention key function; terminals without a PA3 key will have to use the /ATTEN command.

DYNSPC=[NO|YES]

NO

All dynamic disk space areas will be normally started. That is, any files in the extent area of DTSDYNW will be retained ('warm started') and all files in the extent area of DTSDYNC are lost ('cold started').

YES

The type of start (COLD, WARM, NORMAL, BYPASS) for dynamic disk space areas will be read from SYSLOG.

EDFLAG=[73|n]

If you intend to use the editor update flagging facility, specify the column number (1 through 73), where the editor is set to flag changes. The default is column 73. Eight columns are required for the flagging information. This facility is most useful for Assembler source programs. When requested to do so, the editor will flag each record changed with the type of change, the date of the change (MDD) and the ID of the user making the change.

EDEND=[72|80]

The normal editor default for the zone setting is 72 columns. The user may alter the default during an edit session.

IBM recommends that you use the default because a user can always increase (or decrease) the zone according to his editing requirements. However, if you prefer the editor zone to default to all 80 columns for example, specify EDEND=80.

FILEVER=[NO|YES]

NO
No write verify.

YES
Write verify on VSE/ICCF library file.

HCLINE=[132|n|NO]
This operand specifies the physical line size of the hardcopy printer you are going to use for VSE/ICCF message routing. The operand is necessary for IBM 328x printers to avoid additional line spaces. If you use several IBM 328x hardcopy printers with various line sizes you should specify HCLINE=NO. Note, however, that in this situation additional line spaces will occur.

INTCOMP=[NO|YES]
This operand must be specified as YES to include support in VSE/ICCF for the data analysis or TEXT versions of the IBM 3270 keyboard. If you specify INTCOMP=NO, all data analysis special characters entered from the keyboard are interpreted incorrectly.

INTRVAL=[1|2|3]
Specify the timer interrupt interval in seconds. Permissible values are 1, 2 and 3. The default is 1 second. You may specify 2 or 3 which will somewhat reduce processing overhead; however, for certain forms of responses the response time will become worse. For a slow machine, it may be an advantage to use an interval of more than 1 second. For faster machines, however, the default of 1 second should be used.

KATAKAN=[NO|YES]

NO
No hardcopy printer with Katakana feature present.

YES
Hardcopy printer(s) with Katakana feature present; Katakana support requested.

LOADPRT=[NO|YES]
Specify YES or NO to indicate whether or not the optional load protection feature of VSE/ICCF is to be included. You may restrict users from accessing certain programs by specifying their names in the System Program Table (DTSSYSPG). However, this is a job scheduler check and does not prevent a user from writing a program which would internally invoke a restricted phase. When LOADPRT=YES is specified, every fetch or load request is monitored against a user-specified table called the Load Protection Table (DTSSYSLD). If an unauthorized user attempts to access a restricted phase, the access will be terminated, that is, the accessing program is canceled.

Note: If the access control function of the VSE System has been activated by specifying SEC=YES in the IPL command SYS, the load protection function of VSE/ICCF is ignored and the access authorization is done via the access control table; see the section "The VSE Access Control Table (DTSECTAB)" on page 54.

NBUFS=[20|n]
Specify the number of file buffers to be allocated within VSE/ICCF for all library file related activity. The value specified should relate to the number of VSE subtasks (referred to below as NTASKS) and to the average number of VSE/ICCF users that are likely to be logged onto the system at one time. The minimum value is 4 plus NTASKS.

The following table serves as a guide ('T' is the NTASKS value):

No. of Users	Minimum Value	Optimum Value
2	4+T	5+(2*T)
3	5+T	6+(2*T)

No. of Users	Minimum Value	Optimum Value
4	6+T	7+(2*T)
5	7+T	8+(2*T)
6	8+T	9+(2*T)
7	9+T	10+(2*T)
8	10+T	11+(2*T)
9	11+T	13+(2*T)
10	12+T	15+(2*T)
11	13+T	17+(2*T)
12	14+T	18+(2*T)
13	15+T	20+(2*T)
14	16+T	21+(2*T)
15	17+T	23+(2*T)

The specified value is the number of file buffers (in virtual storage) which VSE/ICCF will allocate via GETVIS to service file requests. The more file buffers are specified, the more work VSE/ICCF is able to do in virtual storage without requiring physical input/output.

The fewer file buffers are specified, the more physical input/output VSE/ICCF must do, because more users or execution tasks are attempting to share buffers and buffers must be written and used for new requests. If more buffers were available, data would be retained in storage longer.

The buffer **size** is specified in the tailoring options NRECS (for a CKD device) or CISIZE (for an FBA device).

In determining the maximum usable value of NBUFS, note that VSE/ICCF can use, at the most, 4 buffers per currently logged-on user plus 4 buffers per interactive partition (as defined in the NTASKS= operand, see below) plus 1 buffer.

NRECS=[22|n]

This is the number of 88-byte records per block on the VSE/ICCF library file. This value is required for processing the file when it resides on a CKD device. The value specified must not be less than 8 and not greater than 372. The block size (88 times NRECS) should be efficient for your type of disk device. Consult the track capacity charts for optimum block size.

NUSRS=[30|n]

Determine the maximum number of concurrent execution requests for interactive partitions. If you omit this operand, the default is 30. A good figure to use is 5 plus the expected maximum number of concurrent execution requests. The minimum which may be specified is 6 and the maximum is 64.

NPARTS=[5|n]

Specify the number of interactive partitions to be built for the execution of background jobs. If this operand is not specified, 5 is assumed. The minimum value which may be specified is 3 and the maximum is 35.

A good minimum value to use is 5. The maximum should not exceed the NUSRS value. If most jobs being run in the VSE/ICCF environment are non-conversational jobs, the value specified may be considerably less than the NUSRS value because interactive partitions will be used and then freed for subsequent use quite frequently. However, if several long running conversational jobs will be executed and, thus, tie up interactive partitions for long periods, enough interactive partitions should be specified so that the fast in-and-out user will not be delayed too long by the longer running conversational users.

NTASKS=[4|n]

Specify the number of VSE subtasks allocated to interactive partitions that may be active simultaneously. The minimum value you can specify is 1, the maximum is 10.

Internally, as many VSE subtasks are attached as interactive partitions have work to do, up to a maximum of 25 (31 VSE subtasks may be attached per partition, six of which may be used by VSE/ICCF and CICS Transaction Server). The VSE subtasks are allocated to the interactive partitions for the duration of a job, and they are activated on a time slice basis. Thus, only a restricted number of interactive partitions may run at a time. The other interactive partitions will wait until their associated VSE subtasks got a time slice. Thus NTASKS does not specify the number of VSE subtasks available for interactive execution; it limits concurrent interactive executions for load-leveling purposes.

If more than 25 interactive partitions have to be scheduled, then 25 VSE subtasks are shared by the interactive partitions, on a time slice basis. If less subtasks are available in the system than VSE/ICCF tries to attach, then less subtasks are shared by the interactive partitions.

The generated number of VSE subtasks has an influence on the speed with which jobs are processed in interactive partitions. The more subtasks are generated, the more potential computing resources are applied to interactive partitions. However, specifying too many subtasks may degrade performance.

Some consideration should be given to jobs that are processor intensive and are run in interactive partitions. Assume there are about as many VSE subtasks available as execution requests are scheduled for concurrent background execution. A processor-intensive job then prevents lower-priority batch jobs (in VSE partitions) from executing. It may take processing time away from other background jobs in interactive partitions. To avoid this problem, specify for NTASKS a value smaller than the average number of concurrently active interactive partitions.

The maximum number of subtasks used by the CICS/ICCF partition is

```
NPARTS + 5 + number of VSE subtasks additionally attached
by CICS Transaction Server, up to a value of 31
(NPARTS is a VSE/ICCF tailoring option.)
```

When specifying more than the recommended number of VSE subtasks, be certain to have enough processor storage and processing power available to support the specified level of asynchronous processing.

PGMRINP=[5|n]

PGMRLST=[6|n]

PGMRPCH=[7|n]

PGMRPIN=[8|n]

PGMRLOG=[9|n]

These operands allow you to alter the VSE/ICCF defaults for the SYS numbers associated with interactive partition unit record functions. The default programmer logical unit for reading data or job streams is SYS005; for writing to the terminal user's interactive printer (that is, the user terminal) is SYS006, for writing to the punch area is SYS007, for reading from the punch area is SYS008, and for reading from or writing to the interactive console (that is, the user terminal) is SYS009. If these defaults are not appropriate because of other installation standards, they may be altered using these operands. (If you change the defaults, make sure that the SYS numbers you specified are either unassigned or assigned to real unit record devices when VSE/ICCF is initialized.)

As an example, to change the defaults to SYS013, 14, 15, 16 and 17, you would specify PGMRINP=13, PGMRLST=14, PGMRPCH=15, etc. These assignments may also be temporarily altered during job stream execution via the /ASSIGN statement.

PSIZE=[256|n]

Specify the default size for all interactive partitions. The value specified is the desired storage size divided by 1024. Thus, specifying 160 would cause the size of the interactive partitions to be generated as 160K bytes (160 times 1024). The value specified must be a number from 128 to 1024. If this operand is omitted, 256 is assumed.

The value specified here is used as the size of each interactive partition for which explicit partition information (PARTN and PARTX operands) is not specified.

The interactive partitions are set up dynamically to start at 4K boundaries. The sizes of the partitions must be multiples of 4K. An interactive partition whose size was not specified as a multiple of 4K will be rounded up to a multiple of 4K; its GETVIS area will start at a 4K boundary.

PARTN=(n,xxxx,y,zzzz,n,xxxx,y,zzzz,...)

This operand allows interactive-partition information to be specified on a partition-by-partition basis. Your specification overrides the interactive partition defaults (PSIZE, see the preceding tailoring option).

n =

The interactive partition number, a value from 1 to 35.

xxxx =

The size of the interactive partition in multiples of 1,024 bytes. Specify a value from 128 to 9900. The interactive partitions are set up dynamically to start at 4K boundaries. Moreover, the sizes of the interactive partitions must be multiples of 4K. An interactive partition whose size is not specified as a multiple of 4K will be rounded up to a multiple of 4K; its GETVIS area will start at a 4K boundary.

y =

The number of preallocated work files (IJSYS01 through IJSYS04) present for the partition. For y, specify any of the numbers 0 through 4. A zero indicates that the partition does not have preallocated work files.

zzzz =

A string of one to four alphabetic characters. It indicates the class or classes associated with the interactive partition.

When you specify explicit interactive partition information in this way, all four subfields should be specified. To specify this information for two or more interactive partitions, simply repeat the sequence of the four subfields. Due to an assembler restriction, the PARTN value may not exceed 255 characters. If you need to specify more than 255 characters, use the PARTX operand (see below).

PARTX=(n,xxxx,y,zzzz,n,xxxx,y,zzzz,...)

As indicated for the PARTN operand, approximately 20 interactive partitions fit under that operand. If you need to specify more than that, use the PARTX operand for all partitions larger than the highest partition in the PARTN operand. The PARTX operand has the same syntax as the PARTN operand.

RDR=[FFC|xxx]

RDR2=[FFA|xxx]

PCH=[FFD|xxx]

PRT=[FFE|xxx]

If no real unit record devices are available in the CICS/ICCF partition, dummy devices must be assigned to certain VSE SYS units. If these operands are not specified, dummy devices FFC, FFA, FFD and FFE will be the addresses used. You may specify other unused device addresses if the defaults interfere with actual physical devices on the system or dummy devices generated for VSE/POWER (or other programs).

Note: These dummy devices have to be added during IPL. For performance reasons, you should add the printer as type 'PRT1'.

SPOOL=[250|n]

Specify the default number of print lines to be written to the print buffer area before program deactivation occurs and the printout begins to be transferred to the terminal. If this operand is omitted, a value of 250 is assumed. Specify a value between 50 and 500. The terminal user may alter this default value by issuing the /SET BUFFER command.

If you use IBM 3270 Display Systems, do not specify a value less than 150. 200 should be a good value if you have a mixed network with IBM displays of type 3270, 2740, and 3767. For a system with IBM 3270s only, 250 to 300 is a good range.

TCUPSI=00000000

This operand is not used anymore. It remains as a valid operand to provide compatibility with earlier releases of VSE/ICCF.

TCTOFS=[8|n]

If you have other programs that use the user area in the TCT, you can specify, through this operand, the section of the area which VSE/ICCF is to use. The value of the TCTOFS operand must be a decimal number from 0 to 235. Several other programs which run under CICS Transaction Server use this area for their own purposes. If any of these programs expect to pass values from one transaction to another where a VSE/ICCF terminal session could intervene, make sure that these applications do not use the same area as VSE/ICCF. Examples of such applications are: TEP (CICS Transaction Server supplied or your own), performance statistics, or page copy functions.

TIOA40=[600|n]

TIOA00=[600|n]

These operands specify the size of the TIOA required for writing to a hardcopy terminal such as an IBM 2740, 2741, or 3767 (TIOA40) or to a sequential device such as the system console (TIOA00). The value specified should be numeric and in the range 400® to 1500 for hardcopy terminals (TIOA40) and 400 to 2000 for sequential devices.

Assembling and Cataloging New Tailoring Options

In the preceding section, you reviewed the tailoring options of VSE/ICCF. This section shows the job that you submit to assemble and catalog the tailoring options. Remember that you need not run this job if you are content with the pregenerated options given to you with your z/VSE system.

You specify your tailoring options as operands of the DTSOPTNS macro. The DTSOPTNS macro has the following format:

label DTSOPTNS keyword=option,.....

where 'label' need not be specified. z/VSE provides the job skeleton SKICFGEN in library 59 which should help you code and submit the tailoring job.

Before using the skeleton, copy it to your primary library and edit the copied skeleton.

Figure 2 on page 32 below shows the skeleton. Your VSE/ICCF, as part of z/VSE, is generated with the option values shown in the skeleton. Before changing any option value, carefully consider the impact the change has on the behavior of your z/VSE system.

Comments included in the skeleton are not shown. In the skeleton, each operand is on a separate line. When you edit the skeleton, **do not delete the continuation characters X in column 72**.

The operand in the PHASE statement of the tailoring job can be any name. Likewise, the startup transaction I$ST accepts as operand the phase name of any cataloged generation phase (the default is DTSIGEN). This enables you to generate and work with different VSE/ICCF configurations, depending on your needs.

Note:

1. You can choose any name for the phase to be cataloged. However, one phase with name DTSIGEN must exist because the VSE/ICCF utilities depend on this particular phase.
2. If a phase is generated with a phase name other than DTSIGEN, please ensure that the parameters CISIZE and NRECS are equal to the corresponding values in DTSIGEN. All generation phases must have the same CISIZE and NRECS values.
3. In the figure below, the macro call at the bottom is always for DTSIGEN even if the phase is cataloged under a different name.

```
* $$ JOB JNM=ICCFGEN,CLASS=0,DISP=D
* $$ LST CLASS=Q
// JOB ICCF GENERATION
LIBDEF PHASE,CATALOG=PRD2.CONFIG
// OPTION CATAL
   PHASE DTSIGEN,*                      see Note 1 above
// EXEC ASSEMBLY
         DTSOPTNS ALTSEC=NO,                                           X
               ATN2741=YES,                                            X
               CANKEY=PA2,                                             X
               CISIZE=2048,             see Note 2 above               X
               COMLIB=2,                                               X
               CRJE=(YES,Q,A,D,A),                                     X
               DISPKEY=PA3,                                            X
               DYNSPC=NO,                                              X
               EDFLAG=73,                                              X
               EDEND=72,                                               X
               FILEVER=NO,                                             X
               HCLINE=132,                                             X
               INTCOMP=YES,                                            X
               INTRVAL=1,                                              X
               KATAKAN=NO,                                             X
               LOADPRT=YES,                                            X
               NBUFS=20,                                               X
               NRECS=22,                see Note 2 above               X
               NUSRS=30,                                               X
               NPARTS=5,                                               X
               NTASKS=4,                                               X
               PGMRINP=5,                                              X
               PGMRLST=6,                                              X
               PGMRPCH=7,                                              X
               PGMRPIN=8,                                              X
               PGMRLOG=9,                                              X
               PSIZE=256,                                              X
               PARTN=(1,1024,4,I,                                      X
               2,384,4,A,                                              X
               3,384,4,A,                                              X
               4,512,4,BA,                                             X
               5,512,4,BA)                                             X
               PARTX=,                                                 X
               RDR=FFC,                                                X
               RDR2=FFA,                                               X
               PCH=FFD,                                                X
               PRT=FFE,                                                X
               SPOOL=250,                                              X
               TIOA40=600,                                             X
               TIOA00=600,                                             X
               TCTOFS=8
         DTSIGEN                        see Note 3 above
         END
/*
// EXEC LNKEDT,PARM='MSHP'
/&
* $$ EOJ
```

Figure 2. VSE/ICCF Generation Skeleton SKICFGEN

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.

The cataloged phase DTSIGEN is a table between 100 and 200 bytes long. VSE/ICCF loads this table at initialization, and also when starting execution of a VSE/ICCF utility. VSE/POWER loads DTSIGEN when processing the SLI includes from the VSE/ICCF library file.

Although this discussion pertains primarily to the installation process, you should know that you can tailor your VSE/ICCF at any time after you installed your z/VSE. You only rerun the job ICCFGEN to catalog a new copy of the table DTSIGEN in the VSE sublibrary PRD2.CONFIG. The new copy becomes effective the next time you start up VSE/ICCF.

For modifying options after the VSE/ICCF system has been made operational, note the following: when changing the NRECS or CISIZE option, be certain to back up the VSE/ICCF library file prior to the change. After you cataloged the new options, run DTSUTIL to format your VSE/ICCF library file and to restore it from the backup copy.

Also, before changing CISIZE or NRECS be aware that this number is reflected in the volume 1 label in the VTOC and therefore used at each startup.

Checking CICS Transaction Server Requirements

z/VSE is delivered to you with a pregenerated CICS Transaction Server that fully meets VSE/ICCF's requirements. Refer to z/VSE Planning for information on the CICS Transaction Server tables or consult the *CICS Transaction Server System Definition Guide and Resource Definition Guide.*

This section has some reference information that you might need in special situations. If you plan to use your system in a "normal" fashion, you can ignore this section.

1. *Terminal Definitions*

   SCRNSIZE
   If VSE/ICCF should make use of the alternate screen size of the IBM 3278 display Model 3, 4, or 5, or of the alternate buffer size of the IBM 3287 and 3289 printers, the alternate size of the display or the buffer must be specified.

   UCTRAN
   UCTRAN(YES) should not be specified. UCTRAN(YES) need never be specified for VSE/ICCF terminals since VSE/ICCF performs all of its own translation as specified by the terminal user. (Note that certain other CICS Transaction Server applications running at a terminal also used by VSE/ICCF users may, in fact, depend upon the uppercase translation.)

   If the APLTEXT feature is used for APL terminals, the full screen editor allows the display of the special APL characters.

   IOAREALEN
   Specify a non-zero value.

   For the IBM 3270 Display Systems, specify a value of at least 256 bytes.

   Take into account that the input to the VSE/ICCF full screen editor should fit into the input area.

   Programs running in interactive partitions may use the DTSWRTRD macro. In this case, the IOAREALEN value must be at least as large as the maximum read data length (up to 32,767) minus 4,000.

   USERAREALEN=20+tctofsvalue
   At least 20 bytes of user area must be specified for each terminal entry to be used with VSE/ICCF. The amount of space that user applications take out of the *first portion* of the user area, should be accounted for in the TCTOFS option for VSE/ICCF. For example, if user applications use the first 32 bytes of this area for permanent data or statistics, an offset of 32 should be specified. The distributed VSE/ICCF system uses TCTOFS=8. Therefore, TCTUAL=28 should be specified if you do not plan to change that TCTOFS value.

ATI (YES), TTI (YES)

This must be specified for all VSE/ICCF terminals other than hardcopy (IBM 328x) printers. Hardcopy printers should be specified as ATI (YES) and TTI (NO).

Program Definition in CSD File

- PROGRAM DEFINITION (IESZPPI.Z)

```
Define   Program(DTSPOSTI) Group(VSESPG)
         Language(ASSEMBLER) Reload(NO)
         Resident(NO) Executionset(FULLAPI)
         Execkey(CICS)
Define   Program(DTSICCF) Group(VSESPG)
         Language(ASSEMBLER) Reload(NO)
         Resident(NO) Executionset(FULLAPI)
         Execkey(CICS)
Define   Program(DTSSHUT) Group(VSESPG)
         Language(ASSEMBLER) Reload(NO)
         Resident(NO) Executionset(FULLAPI)
         Execkey (CICS)
```

VSE/ICCF programs use EXECKEY(CICS®) and transactions use TASKDATAKEY(CICS). Thus the storage of VSE/ICCF transaction programs is protected against programs and transactions running in user key.

2. *Transaction Definition in CSD File*

VSE/ICCF makes use of the alternate screen size. Alternate screen size is supported for:

3278, Models 3 through 5.
3287 with an alternate buffer size.
3279, Models 3A and 3B.
3290 with IBM-supplied logical alternate screen sizes.

Alternate screen size is supported for any other terminal if:

- It is compatible to one of the supported terminals as listed above, and
- It has been defined to CICS Transaction Server as the supported terminal.

```
*-----------------------------------------------
      TRANSACTION DEFINITIONS (IESZPCTI.Z)
*-----------------------------------------------
  DEFINE TRANSACTION(iccf)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(ICCF)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$P)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE((1650)    TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$Q)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(DFHCICST)
           TWASIZE(512)      TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$SH)   GROUP(VSESPG)
           PROGRAM(DTSSHUT)  PROFILE(IESXACT1)
           TWASIZE(0)        TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$ST)   GROUP(VSESPG)
           PROGRAM(DTSPOSTI) PROFILE(IESXACT1)
           TWASIZE(0)        TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$1)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$2)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$3)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$4)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$5)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(DFHCICST)
           TWASIZE(512)      TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$6)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$7)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$8)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
  DEFINE TRANSACTION(I$$9)   GROUP(VSESPG)
           PROGRAM(DTSICCF)  PROFILE(IESXACT1)
           TWASIZE(1650)     TASKDATAKEY(CICS)
```

Note:

1. I$ST is the transaction to initialize and start VSE/ICCF. I$ST accepts an operand which is the phase name of a pregenerated VSE/ICCF generation phase. The default name is DTSIGEN.
2. Transaction I$SH forces termination of VSE/ICCF.
3. Transaction ICCF (or iccf) is the first transaction at the start of a VSE/ICCF session.
4. I$$1 is the transaction which is scheduled by CICS Transaction Server when you have pressed ENTER.
5. I$$2 handles the return to the interactive partition after a full screen write.
6. I$$3 and I$$5 handle the display of execution spool print output, the conversational reads, and the full screen write/reads from interactive partitions.
7. I$$6 does the hardcopy printing.
8. I$$8, I$$9 and I$$Q handle the commands /DISCONN DTSFILE and /DISCONN USER.
9. I$$P is the transaction to handle the Interactive Interface of z/VSE.
10. Profile IESXACT1 defines Screensize = Alternate

3. *Destination Control Table (DFHDCT)*

The VSE/ICCF /HARDCPY command allows the user of an IBM 3270 Display System to direct terminal output to a hardcopy printer. This is carried out via CICS Transaction Server transient data queues. There are two ways the VSE/ICCF DCT entry may be coded to provide this support. The way the DCT entry is coded affects the way the /HARDCPY command can be used. Refer to " /HARDCPY" in the VSE/ICCF User's Guide.

The first way a DCT entry can be coded is to define an intra-partition transient data queue with a destination-ID (DESTID) that matches the terminal-ID of a printer terminal. An example of this type of DCT entry is shown below.

```
DFHDCT TYPE=INTRA,
      DESTID=L86P,   PRINTER TERMINAL ID
      TRANSID=I$$6,  VSE/ICCF PRINT TRANS-ID
      TRIGLEV=1      PRINTING IS INITIATED AUTOMATICALLY
```

This type of DCT entry supports the '/HARDCPY devicename' form of the /HARDCPY command.

There are, however, potential problems with this type of DCT entry. If several VSE/ICCF terminal users use this form of the command at the same time their terminal output is likely to become intermingled as it is sent to the transient data queue for printing. Also, this form of the DCT entry will restrict the usage of this destination to the single application (ICCF) indicated by the 'TRANSID' operand. This may conflict with either user written transactions or other IBM licensed programs and FDPs (field developed programs) providing similar hardcopy support functions, for example, DMS/VS.

The second way is to define several 'private' intrapartition transient data queues, one for each VSE/ICCF user or perhaps one for each VSE/ICCF terminal. An example of these DCT entries is shown below.

```
   DFHDCT TYPE=INTRA,
          DESTID=PRTA,  PRIVATE DESTINATION FOR TERMINALA
          TRIGLEV=0     PRINTING IS INITIATED ON EXPLICIT REQUEST
```

```
   DFHDCT TYPE=INTRA,
          DESTID=PRTB,  PRIVATE DESTINATION FOR TERMINALB
          TRIGLEV=0     PRINTING IS INITIATED ON EXPLICIT REQUEST
```

These DCT entries support the '/HARDCPY queuename' form of the /HARDCPY command.

With this type of DCT entry, a user's terminal output is not printed immediately. Instead the user must issue a

```
   /HARDCPY START devicename queuename
```

```
command indicating the private queue to be printed and the printer terminal
terminal on which the output is to be printed.  The '/HARDCPY START ...'
```

command causes VSE/ICCF to request CICS Transaction Server to attach a task to the printer terminal as soon as the printer terminal becomes available. When this new task is attached it reads the 'private' transient data queue indicated in the '/HARDCPY START ...' command and writes the contents on the printer terminal.

These 'private' DCT entries thus avoid the problems associated with the other type of DCT entry. By providing multiple transient data queues, the likelihood of two or more VSE/ICCF users using the same transient data queue simultaneously can be reduced or eliminated. In addition, this type of DCT entry does not conflict with either user written transactions or other IBM program products or field developed programs providing a similar hardcopy function.

4. *System Initialization Table (DFHSIT)*

   - Avoid using the PGCHAIN, PGPURGE, PGCOPY and PGRET operands (BMS paging commands) of DFHSIT. Because a VSE/ICCF user may enter almost any character combination while in input mode, there is no character combination which might not cause difficulties if intercepted by CICS Transaction Server. If you must use one or more of these facilities, try to make the character combinations unique, for example, PGCOPY=..CPY/.
   - Specify 'PA1' or 'PA2' for PRINT= only if absolutely necessary. The specification prevents the use of these keys for VSE/ICCF functions.

     To avoid this conflict and yet obtain the print function, you can use a program with a transaction definition of TASKREQ=PAn. This program could perform the print function, that is, specify PRINT=YES.

5. *Program List Table for Post-Initialization*

   To have VSE/ICCF started during the startup of CICS, the following entry for the VSE/ICCF initalization phase DTSPOSTI must be included in the CICS program list table for post-initialization (DFHPLTPI):

   ```
   DFHPLT TYPE=ENTRY,PROGRAM=DTSPOSTI
   ```

   The z/VSE system already has this setup. Both the DFHPLTPI table and the DTSPOSTI program must be defined in the CSD.

   Note: You can also start VSE/ICCF by calling the transaction I$ST, as explained in "Initialization and Termination Choices" on page 59.

6. *Program List Table for System Shutdown*

   To ensure successful termination of VSE/ICCF via the CICS Transaction Server operator command

   CEMT PERFORM SHUT

   - Add the entries

```
DFHPLT TYPE=ENTRY,PROGRAM=DTSICCF
DFHPLT TYPE=ENTRY,PROGRAM=DTSSHUT
```

to your program list table for system shutdown (PLTSD=... in the DFHSIT) and define your program in the CSD. The entry has to be placed among the entries for the first quiesce stage.

- If you do not specify the above-mentioned entry in your PLT for system shutdown (or if you do not have a shutdown PLT), issue the command /ICCFEND prior to terminating CICS. In case of an immediate shutdown of CICS or if no entry for DTSSHUT exists in PLTSD, VSE/ICCF will be terminated through its own abend routines.

7. *CICS Transaction Server MRO (Multi-Region Operation)*

   CICS Transaction Server MRO allows the operation of multiple-connected CICS Transaction Server systems in the same CPU. This allows data, terminals, and transactions to be shared across the connected systems. VSE/ICCF may run together with one of the connected CICS Transaction Server systems in one partition. CICS Transaction Server resources used by VSE/ICCF (transactions, temporary storage, transient data, and so on) must be local to the CICS Transaction Server system running with VSE/ICCF. Logon to CICS Transaction Server can, however, be done from terminals in both CICS Transaction Server systems.

8. *Miscellaneous*

   To avoid the loss of information when a hardcopy printer runs out of paper, make sure that the transaction for this printer is terminated and the printer is placed out of service. You do this:

   - Issue the master terminal transaction CEMT...

   After the printer has been made ready, it must be placed back in service via the CEMT transaction.

   To work with the Katakana feature of an IBM 328x hardcopy printer, you must request Katakana support by means of the VSE/ICCF tailoring option KATAKAN (see "Description of VSE/ICCF Tailoring (DTSOPTNS) Options" on page 24). If Katakana support is requested, an extra byte in the print buffer will be reserved for each NL and EM character. As a result, the buffer for a hardcopy printer without the Katakana feature will not be completely utilized. Output will, however, be corrected. Within CICS Transaction Server nothing special has to be provided.

Checking the Library File Allocation

z/VSE places the VSE/ICCF library file on SYSWK1 under the file ID 'ICCF.LIBRARY'. Layout and allocations on SYSWK1 for the various disk device types are shown in z/VSE Installation.

If the given layout or allocation does not fit your needs, you may want to come up with a new structure of your library file. Answer for yourself the following questions:

1. How much space should be allocated?
2. Where should the file be placed in relationship to the page data set and to work files?

How Much Space to Allocate

The amount of space to allocate is calculated in terms of 88-byte records. The number of 88-byte records to be allocated may be approximated using the following formula:

```
  1000 + (n x d x m) + (u x 5000)
where  n = The number of user profiles which have been defined.
       d = The average number of members which a given user will
           have in his library.
       m = The average size of a library member.
       u = The number of users which may be logged on at any one
           point in time.
```

For example, assume a system defined with four terminals (u=4) and ten user profiles (n=10). Suppose also that the average number of members which a given user will have stored at any one time is 100 (d=100) and the average size (in records) of a member was 150 (m=150) records. Then the estimated file size would be:

```
1000 + (10 x 100 x 150) + (4 x 5000) = 171,000 records
```

Now this value must be translated into disk space. Suppose the number of records per block on the VSE/ICCF library file is 22 (NRECS=22). Thus, the block size is 22 x 88 = 1936 bytes. On an IBM 3380, for example, 19 blocks of this size fit on a track. Thus, 19 times 22 or 418 records will fit on a track. Therefore, 171,000/418 or about 410 tracks of 3380 space are needed to store the file.

Where to Place the File

It is important that the VSE/ICCF library file is placed in a suitable relationship to the page data set and the compiler work files. It cannot reside in VSAM-managed space and it cannot be shared across VSE systems. See the section "Optimizing Performance" on page 55 for more information concerning file placement.

However, assume for the moment that we decide to place the VSE/ICCF library file (DTSFILE) on a disk which has serial number SYSWK1.

Defining the Library File (DTSFILE)

Now DLBL/EXTENT information for DTSFILE can be defined. It should be added to the VSE standard label information area as follows:

```
// JOB STD LABELS
// OPTION STDLABEL=ADD
// DLBL DTSFILE,'ICCF.LIBRARY',99/365,DA
// EXTENT SYS010,SYSWK1,1,0,nnnn,420
/&
```

If multiple volumes are to be used for the VSE/ICCF library file, the SYS numbers for the extents must be in ascending and consecutive order for each different volume. It is necessary that the first extent of the file corresponds to the lowest SYS number. For example:

```
// DLBL DTSFILE,'ICCF.LIBRARY',1999/365,DA
// EXTENT SYS010,SYSWK1,1,0,nnn,150
// EXTENT SYS011,SYSWK2,1,1,nnn,150
// EXTENT SYS011,SYSWK2,1,2,nnn,300
```

Note that the first extent is on one drive while the second two are on another drive. If other than the SYS numbers in the IBM-supplied initialization job stream are used for DTSFILE, you must provide appropriate assignments during initialization of VSE/ICCF.

z/VSE provides job skeleton SKDTSEXT which has statements for defining the VSE/ICCF library file DTSFILE. The skeleton has a more general purpose: to extend the VSE/ICCF DTSFILE to a multi-volume, multi-extent file. The skeleton is shipped in library 59. Figure 3 on page 39 shows the skeleton and the variables you must specify.

To extend the DTSFILE, you must perform the following steps:

1. Back up 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 3 on page 39 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. Shut down the system and perform a MINI startup. Release the jobs in the reader queue. First release the extend job and then its successful completion the restore job.

6. To activate the changed characteristics of the DTSFILE, shut down the system and perform a normal startup.

```
* $$ JOB JNM=ICCFEXT,CLASS=0,DISP=D
* $$ LST CLASS=A,DISP=D
       .
       .
// JOB ICCFFORM  FORMAT YOUR  VSE/ICCF DTSFILE ON NEW EXTENTS
// DLBL DTSFILE,'ICCF.LIBRARY',1999/365,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(99) USERS(99)
/*
/&
// JOB UPDATE    UPDATE STDLABEL AREA  AND DTRICCF.PROC
// OPTION STDLABEL=DELETE
DTSFILE
/*
// OPTION STDLABEL=ADD
// DLBL DTSFILE,'ICCF.LIBRARY',1999/365,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 3. 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. You should make sure that a definition does not cause an overlap on other system files.
- 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 the VSE/POWER and CICS/ICCF partitions.

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
```

Building a New VSE/ICCF Library File

Based on the considerations of the preceding step, you may now want to build a new VSE/ICCF library file.

The utility program DTSUTIL, which is described in section “Library File Maintenance (DTSUTIL) Utility Program” on page 123 is used to initially build the total VSE/ICCF library file, depending on the specification of the LIBRARIES and USERS operands in the FORMAT control statement. An **individual** library within the total VSE/ICCF library file is added via the ADD LIBRARY control statement, a user is added via the ADD USER control statement.

Creating Library and User Profile Records

Before you start building your VSE/ICCF library you should ask yourself the following questions:

- How many potential libraries should be defined?
- How many actual libraries should be defined?
- How many potential user profiles should be allocated?
- How many actual user profiles should be defined?
- How are the users to be named?
- What passwords and profile characteristics should be given to individual users?

Once these questions have been answered, you should be able to define and initialize the library file. Let us now look at these questions one by one.

How Many Potential Libraries? When the VSE/ICCF library file is initially formatted, you must specify the number of potential libraries to be allocated. The specification should be larger than the anticipated number of libraries to allow for future library expansion without having to reformat or restore the file.

If you can foresee a need for 10 libraries when you initially build the file, you should probably specify 20 to allow for more actual libraries to be added later. The number of potential libraries is specified via the DTSUTIL FORMAT command. This specification can be increased by reformatting the VSE/ICCF library file or by restoring the file from a backup copy (for a description of the FORMAT command refer to “FORMAT Command” on page 125).

How Many Actual Libraries? The actual number of libraries to be defined depends on the users' requirements. (Keep in mind that z/VSE reserves a few libraries for its own use.) If, for example, 10 user profiles will be entered into the system, no more than 10 actual libraries would be needed unless all users had their own libraries and some users owned more than one library. If each user owned a library, or several libraries (which is the most efficient allocation scheme), there would be one or more actual libraries (header records) for each actual user profile defined. If several users will share a single library, however, fewer actual libraries than actual users will probably be needed. A library is actually defined by the ADD LIBRARY command of DTSUTIL.

How Many Potential User Profiles? When the VSE/ICCF library file is first formatted, a number of potential user profiles must be defined. However, actual user profiles need not be specified for all potential profile record allocations. The profile record positions allocated but not defined allow room for expansion.

Assuming that at present you have 14 actual user profiles to be defined, you should probably specify a larger number (for example, 25) as the number of potential profiles. This will allow 11 profiles to be added later. The number of potential user profiles is defined via the FORMAT command of DTSUTIL. This specification may be altered during a FORMAT operation or during a library file RESTORE operation.

How Many Actual User Profiles? A user profile is actually defined via the ADD USER command of DTSUTIL. z/VSE provides two convenient means for adding, changing or deleting user profiles:

1. The *Maintain User Profiles* dialog. How you work with the dialog is described in z/VSE Administration.
2. The batch program IESUPDCF. Its use is described in z/VSE System Utilities.

The requirements of your installation determine how libraries and user profiles are allocated. A user identification (together with its user profile) may represent an individual or a group or a project. If the user profile pertains to a group, an individual at different times may use the system under more than a single user ID if this individual in fact belongs to multiple groups. However, bear in mind the recommendations given in the section “Optimizing Performance” on page 55. Note that any user may be logged onto the system only once with the same user ID.

What Are the Password and Profile Characteristics ? Each user whose profile is added to the VSE/ICCF library file will be represented by a unique 4-character identification code which the user must specify when logging on. The codes are defined by the installation. They may relate to the user in a mnemonic or functional manner but this relationship is not required. In addition a logon password and user characteristics must be defined for each user identification. This information is defined in the ADD USER command of the DTSUTIL utility. For the format and options of the ADD USER command, refer to the description of the DTSUTIL utility. Again, you get help in making these specifications through the "Maintain User Profiles" dialog described in z/VSE Administration. Or, you can use the batch program IESUPDCF described in z/VSE System Utilities.

The VSE/ICCF administrator should have the highest possible access levels as defined by the OPTA, OPTB and OPTC operands of the ADD USER control statement. In most examples given in this documentation, the VSE/ICCF administrator works with user code AAAA and library 1.

The ADD commands for the LIBraries and USERs can all be made at this point or some may be deferred until later if the actual users' requirements are not yet known. The important point is that the FORMAT command should be specified large enough to cover all users and libraries which the installation might want to add in the near future. If the FORMAT factors are later found to be too small, the file must be backed up and restored using the DTSUTIL utility program. The new factors may be specified on the RESTORE command in order to increase library and/or user allocations.

z/VSE has in library 59 the job skeleton SKICFFMT for building the VSE/ICCF library file (DTSFILE), or for reformatting it. Figure 4 on page 42 shows the skeleton's major statements. Comments which appear in the skeleton are not shown here. The ADD LIBRARY commands reflect the structure of the library file of the delivered z/VSE.

Refer to Figure 5 on page 43 for a series of ADD USER commands. They are used by z/VSE to set up its predefined users such as SYSA, PROG and OPER. You may also consider z/VSE's restrictions on user profile options as described in “Special z/VSE Considerations for Adding a User” on page 135.

The DTSFILE generated by z/VSE defines 199 libraries and 199 VSE/ICCF user ID records. You can, however, reformat the DTSFILE to create up to 9999 user ID records and up to 9999 libraries. The z/VSE Interactive Interface imposes these limits; VSE/ICCF itself supports up to 32767 users and libraries.

If you use the skeleton, first copy it from VSE/ICCF library 59 to your primary library. Then edit the copied file.

Note:

1. If you change the number of libraries or users in the FORMAT statement, you must supply the same information on the RESTORE statement of the job used to restore the VSE/ICCF libraries after the reformat run. If you want to change an existing VSE/ICCF library, use the ALTER command of DTSUTIL.
2. For z/VSE, you must define VSE/ICCF libraries with the DATE option.

Each section of the skeleton is shown in separate parts of the figure. A description of job or DTSUTIL statements follows each part of the figure.

The skeleton 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
// EXEC DTSUTIL
FORMAT LIBRARIES(99) USERS(99)
```

Figure 4. Formatting the VSE/ICCF DTSFILE (Skeleton SKICFFMT)

In the ASSGN statement, replace the variable **-V001-**. Specify the volume number of the disk where the DTSFILE resides.

```
*  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
```

The skeleton adds the following libraries:

1   = For VSE/ICCF administrator
2   = Common library
3-6 = Public libraries

```
*  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
```

Libraries 7 - 49 are private libraries that can be assigned to users.

Note: Libraries 8, 9, 10 and 11 are used as primary libraries by the z/VSE supplied users OPER, PROG, SYSA and $SRV.

```
*  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
```

The skeleton adds libraries 50 - 68. They are reserved for z/VSE and are used by the Interactive Interface.

```
*  ADD LIBRARIES 69 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
```

The skeleton adds libraries 69 - 199. They 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.

The following figure shows examples of adding user profiles. These commands are also used by z/VSE to establish its predefined users.

```
ADD USER ID(AAAA) LIBRARY(1) PASSWRD(ICCF)
ADD USER ID(AAAA) MAXPRINT(3000) MAXPUNCH(3000) MAXSTATE(5000)
ADD USER ID(AAAA) OPTA(01110001) OPTB(11111010) OPTC(01000000)
ADD USER ID(AAAA) LOGONRTN(@L$LOGON)
ADD USER COPY(AAAA) ID(POST)  PASSWRD(BASE) LIBRARY(10)
ADD USER COPY(AAAA) ID(ADMN)  PASSWRD(ADMNPW) LIBRARY(67)
ADD USER COPY(POST) ID(SYSA)  PASSWRD(SYSA) LOGONRTN(@L$LOGON)
ADD USER ID(AZZZ) LIBRARY(2) PASSWRD(AZZZ)
ADD USER ID(AZZZ) MAXPRINT(2000) MAXPUNCH(2000) MAXSTATE(2000)
ADD USER ID(AZZZ) OPTA(01110001) OPTB(11111010) OPTC(01000000)
ADD USER ID(AZZZ) LOGONRTN(@L$LOGON)
ADD USER ID(PROG) LIBRARY(9) PASSWRD(PROG) LOGONRTN(@L$LOGON)
ADD USER ID(PROG) MAXPRINT(3000) MAXPUNCH(3000) MAXSTATE(3000)
ADD USER ID(PROG) OPTA(00000100) OPTB(10000000) OPTC(01000000)
ADD USER COPY(AAAA) ID(OPER)  PASSWRD(OPER) LIBRARY(8)
ADD USER COPY(OPER) ID($SRV)  PASSWRD($SRV) LIBRARY(11)
```

Figure 5. Examples of ADD USER Command

Adjusting Startup JCL

An example of a startup job stream is shown in "Initialization Job Stream" on page 60. The job stream shown in that section is quite comprehensive in that it touches upon all aspects of a VSE/ICCF startup job control. It can thus form the basis for your own VSE/ICCF initialization deck. But be aware of the VSE/ICCF startup job that is delivered with z/VSE. It may well suit your requirements so you would not have to develop a new startup JCL. The z/VSE delivered startup job is shown in Figure 12 on page 69.

The following discussion concentrates on the general job stream, not so much on the one delivered with z/VSE. But this information is useful, if you ever want to modify the z/VSE delivered job stream.

For the startup job consider the following modifications:

1. Making proper assignments.
2. Removing or adding pre-allocated work file DLBL/EXTENT statements.
3. Removing or adding dynamic space area DLBL/EXTENT statements.
4. Altering EXTENT statement information.
5. Inserting DLBL/EXTENT statements for permanent files.

Making Proper Assignments

The // ASSGN statements in the sample initialization JCL should be adjusted to your disk configuration and file placement. SYS000 through SYS004 may be assigned to the disk device(s) where the corresponding work files (IJSYS00 through IJSYS04) will be placed. In any case, SYS001, 2, 3 and 4 must be assigned to some disk devices.

The following logical units can be assigned only if your job is processed in a partition that owns corresponding actual (not dummy) unit record devices:

System logical units SYSIPT, SYSLST, and SYSPCH.

Programmer logical units SYS005 through SYS007 (or your equivalent of SYS005 through SYS007 as specified in the PGMRINP, PGMRLST and PGMRPCH tailoring options).

For example, when using a partition which does not own actual unit record devices, do not assign these units: VSE/ICCF will make its own assignments to the dummy devices.

If logical units for unit record devices are assigned IGN or UA, VSE/ICCF assigns these units to the dummy devices as defined by VSE/ICCF tailoring options.

SYS008 (or the equivalent specified as PGMRPIN) should always be assigned as UA.

SYS009 (or the user equivalent specified as PGMRLOG) must be assigned to SYSLOG.

It is recommended that SYS010 through SYS01n be assigned in ascending sequence to the disk units on the system accessible by VSE/ICCF: SYS010 to the first unit (for example 130), SYS011 to the second unit (131).

Keep in mind the restrictions that some compilers place on the assignment of programmer logical units. Refer to the note within the discussion of the PGMRINP operand on page “PGMRLOG=[9|n]” on page 29.

Removing or Adding Pre-Allocated Work File DLBL/EXTENT Statements

The example of startup JCL contains pre-allocated work files for the first two interactive partitions.

Each work file is represented by a DLBL statement with the file name IKSYSpn where p is the interactive partition number (1 through 9 and A through Z) and n is the number of the work file (0 through 9). If your system will use dynamically allocated space for the work files IJSYS01 through IJSYS04 for all interactive partitions, the DLBL statements for the pre-allocated work files (IJSYS01 through IJSYS04) will not be used.

If pre-allocated work files for IJSYS01,2,3 and 4 (or 1,2 and 3 or 1 and 2) are available for an interactive partition, VSE/ICCF should be made aware of this fact so that it can ignore any dynamic allocation requests for these files. VSE/ICCF is made aware of the existence (and number) of pre-allocated work files for an interactive partition via the PARTN and PARTX tailoring options. **For each pre-allocated work file as defined in the PARTN/PARTX tailoring option, the associated labels must be present**.

The example of startup JCL assumes that the pre-allocated work files reside in VSAM-managed space. You can also define them as non-VSAM sequential files.

Removing or Adding Dynamic Space Area DLBL/EXTENT Statements

Dynamic space is disk space from which VSE/ICCF will allocate tracks or blocks dynamically based on space requests from interactive partition executions.

There are two types of dynamic space: DTSDYNC which denotes the *cold start* dynamic space and DTSDYNW which is the file name of the *warm start* dynamic space (see section “12. Define Dynamic Space Areas” on page 62). In your installation you may define cold start and/or warm start dynamic space. Each dynamic space type is defined by a set of one // DLBL and one or more // EXTENT statements. The // EXTENT statements define the disk areas from which the dynamic space is built.

The maximum size of any given disk area is 7200 tracks or 115200 blocks. The total number of EXTENT statements for both DTSDYNC and DTSDYNW may not exceed nine. However, the 1 through 9 disk areas may be on the same volume or on different volumes. All DTSDYNC and DTSDYNW extents allocated to one specific volume must have the same SYSnnn. If, for example, you have a cold start dynamic space consisting of one disk area on volume 111111 and a warm start dynamic space consisting of 2 disk areas on volume 111111 and volume 222222 you would have to code:

```
// DLBL DTSDYNC
// EXTENT SYSxxx,111111,,,start,length
// DLBL DTSDYNW
// EXTENT SYSxxx,111111,,0,start,length
// EXTENT SYSyyy,222222,,1,start,length
```

Inserting DLBL/EXTENT Statements for Permanent Files

If you want to allow terminal users to access certain permanent files, place the DLBL/EXTENT statement sets for these files in the startup job stream or in the VSE standard label area. This will mean that users accessing these files will not have to specify file information (/FILE statements) in their interactive partition job streams. If terminal users will access ISAM files or any other multiple extent files from interactive partition executions, they are not accessible via file information (/FILE statements) provided in the interactive partition job stream. The DLBL/EXTENT statement sets for these files must be placed in the startup job stream or in the VSE standard label area. The proper assignments (ASSGN) must be made in the startup job stream.

Altering the VSE/ICCF Configuration

To alter the VSE/ICCF configuration you must prepare and submit a tailoring job, as discussed at the beginning of this chapter. Reconfiguration via SYSIPT or system console (which was possible in earlier releases of VSE/ICCF) is no longer supported.

Installation of Compilers

Compilers can be installed, if they need to be installed at all, anytime after VSE/ICCF is operational.

For performance and convenience reasons, install the compilers in accordance with the guidelines specified in this section.

Assembler

The Assembler for VSE is part of your z/VSE. Therefore, no installation effort is required.

The current version is the High Level Assembler for VSE. Any call of the form

```
/LOAD ASSEMBLY
```

within your job streams will automatically be interpreted as

```
/LOAD ASMA90
```

with

```
PARM='SIZE(MAX),CPAT(SYSL),EX(LBX(EDECKXIT)),FOLD'
```

To exploit the full High Level Assembler instruction set

```
/LOAD ASMA90
```

must be specified in the job stream or in the Assemble procedure. The High Level Assembler is shipped with procedure ASMARUN . This procedure resides in the High Level Assembler product library. Also ASSEMBLE procedure in VSE/ICCF library 2 could be modified. Refer to the *IBM High Level Assembler for VSE, Programmer's Guide* for recommended settings.

Note: Samples throughout the VSE/ICCF literature might still show:

```
// EXEC ASSEMBLY
```

```
or

/LOAD ASSEMBLY
```

For information on the Assembler, refer to the *IBM High Level Assembler for VSE, Programmer's Guide*.

VS FORTRAN

VS FORTRAN is not an optional program of z/VSE. Therefore, the simplified installation procedure of z/VSE does not apply. Nevertheless, dialog support for installing this type of licensed program is available as described in z/VSE Installation.

RPG II

DOS/VS RPG II is an optional program of z/VSE. To install it, use z/VSE's simplified installation of optional programs as described in z/VSE Installation.

Miscellaneous Tailoring Tasks

The tailoring tasks described here should be done prior to starting VSE/ICCF operation.

Modifying SUBMIT, RELIST, GETL, GETP, and GETR

When using the VSE/POWER submit-to-batch facility under VSE/ICCF, the SUBMIT, GETL, GETP, and GETR procedures and the RELIST macro should be adapted to your standards, such as job and output classes. They may also be set up so that certain JECL functions are not available to the terminal user. For example, you may want all submitted jobs to be forced into a certain class so that they are all executed in a certain VSE partition.

Be aware that z/VSE's Interactive Interface provides comfortable means for submitting jobs to VSE/POWER and for managing VSE/POWER batch queues.

If you want to adapt the SUBMIT procedure to your own defaults and restrictions refer to section “Submit to Batch (DTSSUBMT) Program” on page 83.

Setting Hardcopy Terminal Identification

The VSE/ICCF-supplied macros HC and PRINT issue the command /HARDCPY ON. The ON operand implies that the user has previously made the relevant hardcopy terminal known to VSE/ICCF via a command in the form /HARDCPY terminal-id.

You can alter the macros so that they actually specify the hardcopy terminal id instead of ON. If different users will use different hardcopy terminals, it is possible to have different versions of these macros specifying different hardcopy terminals in the appropriate users' libraries.

Setting Forced-Logoff Save Areas

If the proper user-related save areas have been established, VSE/ICCF will make sure that user data is not lost should a forced logoff occur. For example, if a member called IN$$xxxx (where 'xxxx' is the userid) is in a library that is accessible when the forced logoff occurs, the data in the input area for the user will be saved in this member. When the user logs back on, IN$$xxxx can be inserted into the input area.

Thus, each user who is to have this capability needs a dummy member called IN$$xxxx in his library, where 'xxxx' is the userid. This library must be accessible by the user when the forced logoff happens. Therefore, the preferred library for these members is the common library.

This facility may also be used for saving user log file areas (LG$$xxxx) or punch file areas (PC$$xxxx) should a forced logoff occur.

Getting Started

You may at any time learn to work with VSE/ICCF by performing the following steps:

- Log on to the system.
- Execute a sample terminal session.

Access Control Facilities

VSE/ICCF provides its own protection scheme through security levels in user profiles and through the following access control tables:

System File Table
System Program Table
Load Protection Table

For files listed in the System File Table, an access violation can be detected under VSE/ICCF only when referencing the file via a /FILE statement. It is thus not possible to detect access violations if the label information for a file has been provided during the initialization of VSE/ICCF or if it is in the standard label area. Programs listed in the System Program Table are rejected when invoked by a /LOAD, a /RUN or a $ statement. Phases listed in the Load Protection Table are rejected when the corresponding FETCH/LOAD is executed.

The VSE/ICCF access control tables are phases in the VSE system sublibrary (IJSYSRS.SYSLIB) and are loaded during initialization. As they also contain information that is not security related, they must exist even if you do not use the access control of VSE/ICCF.

VSE/ICCF as delivered to you contains the three tables as source books and as phases with predefined values in the VSE system sublibrary (IJSYSRS.SYSLIB). You may change the predefined tables by assembling and cataloging your own version of these tables. The following sections describe how you do that.

VSE/ICCF will not use its own access control tables if the access control function of the VSE System is active. This function is activated through the IPL SYS parameter SEC. How the VSE access control function helps you to protect your VSE/ICCF resources is described in section “The VSE Access Control Table (DTSECTAB)” on page 54.

System Program Table

The System Program Table contains an entry for each program (compiler, utility, application program, and so on) that is to receive **special treatment** under VSE/ICCF.

Each program name entered in the table must be a phase in a VSE sublibrary. This sublibrary must be part of the CICS/ICCF partition's search chain that is defined in the startup job stream for that partition. In other words, the sublibrary's name must appear in the SEARCH operand of a

```
// LIBDEF PHASE,SEARCH=...
```

statement during startup of CICS with VSE/ICCF. Also, this phase must be invoked via an explicit or an implicit /LOAD. If the phase is fetched or loaded by another phase, then it will **not** be checked against the System Program Table.

Program Characteristics

The characteristics that can be given to phase names in this table are described below. Each characteristic is represented by a 'flag' name which is used on the table entry definition to specify the characteristics which are to apply to the phase.

- Compiler

Any compiler which produces standard VSE object programs (ESD, RLD, TXT) on SYSPCH should be flagged in this table. This causes VSE/ICCF to implicitly invoke the LINKNGO program to process and execute the resulting object programs. The flag name for this characteristic is RQEOPCMP.

- Authorized Program

  It may be desirable for the installation to restrict the use of certain programs by unauthorized users (to avoid misuse of, for example, VSE/DITTO by terminal users). These programs and others may be specified as authorized programs or functions and, thus, restricted for use by authorized users only. To use such a program, the user requesting it must have the authorized user flag (bit 4 of OPTB in the user profile) set on. The flag name for the authorized program characteristic is RQEAUTH.

- No Rollout

  It may be desirable to prevent certain programs from becoming dormant when their time slice elapses. For example, very short running job steps which perhaps tie up some critical system resource might be given the no-rollout characteristic. This would allow the program, once it was started, to finish without being made dormant because of a time slice elapse. For very short running programs, it may be more efficient to let the step complete rather than to have to page out all of its storage and then page it back in when it receives another time slice.

  Another possible use for this characteristic is with conversational programs. A conversational program normally requests input from a terminal, processes the input, writes a message back to the terminal and then requests input again. Normally, this processing time does not exceed a single time slice. However, in cases where the processing time occasionally exceeds the time slice (5 seconds) or always exceeds the time slice by only a small amount, it may be of advantage to let the processing cycle complete rather than have the program paged out and then back in again when it becomes eligible for another time slice.

  In summary, the use of this characteristic should improve system performance and terminal response for conversational programs as long as the processing portion of the read-process-write-read cycle is never more than about two times the slice value (of 5 seconds). The flag name for this characteristic is RQEOPNOR.

- Priority Program

  You can make certain program phases *priority* programs, which means that these programs will be given a time slice before a non-priority program which might also be waiting. That is, when the high priority task scans the system for interactive partitions which require time slices, it always checks for priority programs first.

  This practice is useful when more than one task (NTASKS=) has been specified and it will generally be used for conversational type programs. In order to give a terminal user of a conversational program better response, it is wise to flag the conversational program as a *priority program*. This will improve response at the terminal itself and should not adversely affect throughput of other background jobs since conversational programs are usually dormant longer while the terminal user keys in a response. The flag name for this characteristic is RQEPRTY.

- Non-Concurrent Usage

  You may have programs that should not be running concurrently in different interactive partitions because, for example, they might update the same file at the same time without locking the file while updating. This means that unpredictable results could occur. However, you can overcome this problem by coding such programs into the System Program Table with one of the three non-concurrent use (ENQ) flags set on. These flags are RQEOPSRC, RQEOPRLO and RQEOPCIL.

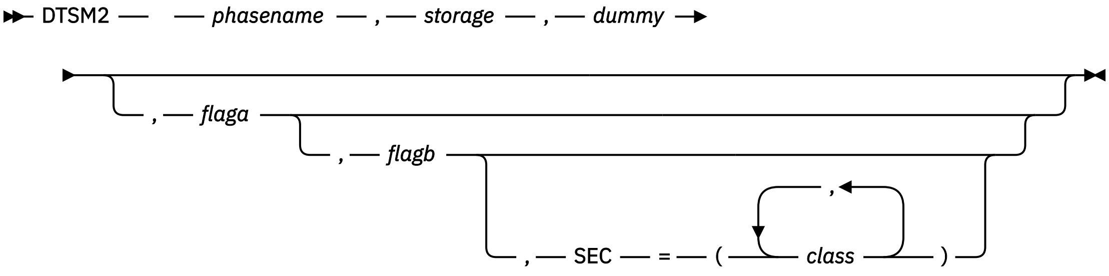
  If a job running in an interactive partition requests the execution of a program with one or more of these flags set, the job will only be executed as long as there is no execution in any other interactive partition which has also set the same flag or flags.

- Program Protection

  If a program may be executed only by a certain class or level of users, this fact may be indicated by the SEC= operand. A user may have no security class (0) or a class 1 through 32 as indicated in his profile. If the user's class matches any of the classes specified for the program (assuming that the program is secured), access to the program will be allowed. For example, SEC=(2,5,6,32) indicates that the program may be executed only by a user whose security class or level is one or a combination of 2, 5, 6, and 32.

Define a System Program Table Entry

A System Program Table entry is defined by the following statement:



Note: Each table entry is 16 bytes long.

phasename
Is the name of the program phase that is cataloged in a VSE sublibrary. It is the same name as is specified on the /LOAD statement which is used to invoke the program.

storage
Defines the maximum amount of virtual storage which the program will be able to use. This operand is used to force a program to run with less virtual storage than the entire interactive partition size. It allows you to restrict storage usage in programs such as compilers which would normally dynamically allocate the entire remaining interactive partition as buffer space, which would affect performance negatively. To find out about the minimum storage requirements of the various IBM compilers, refer to the publications in which the compilers are documented.

The value entered for 'storage' should be the desired storage value divided by 1024. Thus, specifying 54 would cause 54K or 54 times 1024 to be made available to the program. Specify 0 if you do not want to define a maximum. This implies that the entire interactive partition will be used.

The storage value specified here can be overridden through the GETVIS option of the /OPTION statement.

dummy
Should be specified with the same value as the 'storage' operand.

flaga
Specifies the special characteristics to be applied to the program. If no characteristics apply (for example if you only want to restrict storage), specify this operand as the digit 0 or not at all. Otherwise, specify the characteristic to be applied. If more than one characteristic is to be applied to a program, the flag names should be separated with a plus sign (for example, RQEOPNOR+RQEPRTY). The possible flags are RQEOPNOR, RQEPRTY, RQEOPCMP and RQEAUTH.

flagb
Specifies the non-concurrent use flags; specification of this operand is optional. If the program being entered in the table does not require single-thread execution, this operand should be omitted or specified as '0'. The possible flags which may be specified are RQEOPRLO, RQEOPSRC and RQEOPCIL. Multiple flags may be specified as in 'flaga'. An explanation of the flags and their names is given above, in section "Program Characteristics" on page 47.

SEC
Specifies the security class of the program. If the operand is omitted, security class 0 (no security) is assumed. You may specify any number(s) from 1 to 32. For an explanation of program security classes, refer to the last paragraph in section "Program Characteristics" on page 47.

Examples

1. Define a System Program Table entry for the program phase SCHEDULE which requires the priority program and no roll out characteristics. The storage required is set to zero which implies use of the entire interactive partition.

   ```
   DTSM2 SCHEDULE,0,0,RQEOPNOR+RQEPRTY
   ```

2. Restrict the use of the program DITTO to authorized users:

   ```
   DTSM2 DITTO,0,0,RQEAUTH
   ```

3. Restrict the amount of storage used by the program FINESORT to 40K.

   ```
   DTSM2 FINESORT,40,40,0
   ```

4. Restrict the program named ENGSTRT to users with security classes 4, 5 or 8.

   ```
   DTSM2 ENGSTRT,0,0,0,0,SEC=(4,5,8)
   ```

Example for Defining an Entire System Program Table

Figure 6 on page 50 shows an example of a System Program Table. This example may differ from the table delivered with VSE/ICCF with regard to the actual storage values. To find out about the currently valid storage requirements, refer to the latest documentation of the product in question.

The System Program Table delivered with VSE/ICCF resides in the VSE system sublibrary (IJSYSRS.SYSLIB) as source member DTSSYSPG.A. It resides in this sublibrary also as member DTSSYSPG.PHASE.

```
   DTSM2 ASSEMBLY,48,48,RQEOPCMP
   DTSM2 VFORTRAN,880,880,RQEOPCMP
   DTSM2 DTSOBJCT,24,24,RQEOPCMP    ICCF OBJECT UTILITY
   DTSM2 RPGII,72,72,RQEOPCMP       RPG II COMPILER
   DTSM2 DTSUTIL,0,0,RQEAUTH
   DTSM2 DTSANALS,0,0,RQEAUTH
   DTSM2 SORT,40,40,0
```

Figure 6. Example for Defining a System Program Table

Modifying the System Program Table

To modify the System Program Table, run the job SYSPG, which is shown in Figure 7 on page 50. You either code all DTSM2 statements, repeating those in the IBM-supplied source book DTSSYSPG.A (possibly with modifications) and adding your own. Or you create a copy of the IBM-supplied source book DTSSYSPG.A and modify or add DTSM2 statements in that copy. In the job stream below, your only input would then be a 'COPY name' statement instead of all the individual DTSM2 statements ('name' being the name that you chose for your copy).

After completion of this job, a new copy of the phase DTSSYSPG.PHASE is cataloged in the VSE system sublibrary (IJSYSRS.SYSLIB). If VSE/ICCF is running at the time the assembly and link edit are performed, the new version of the table will not take effect until the next time VSE/ICCF is started.

```
// JOB SYSPG
// LIBDEF PHASE,CATALOG=IJSYSRS.SYSLIB
// OPTION CATAL
   PHASE DTSSYSPG,*
// EXEC ASSEMBLY
         START 0
         DTSM2 ASSEMBLY,...
         . . .
         DTSRQED
         END
/*
// EXEC LNKEDT,PARM='MSHP'
/&
```

Figure 7. Example Job - Modify the System Program Table

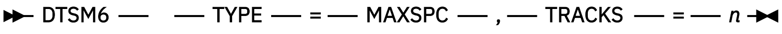
System File Table

The System File Table serves three major purposes:

- Setting the maximum single dynamic space allocation.
- Defining on which volumes VSE/ICCF users can define normal VSE files.
- Restricting the use of certain files to authorized users or users of a given security class. An authorized user is a user with bit 1 set on in OPTB of the user profile. The user may access VSE volumes and files regardless of security classes or restrictions to authorized users.

Define Maximum Dynamic Space Allocation

There can be only one entry in the System File Table defining the largest single dynamic disk space allocation. If your dynamic file space request exceeds this value, the request will be altered to the table value. The table entry is defined as follows:

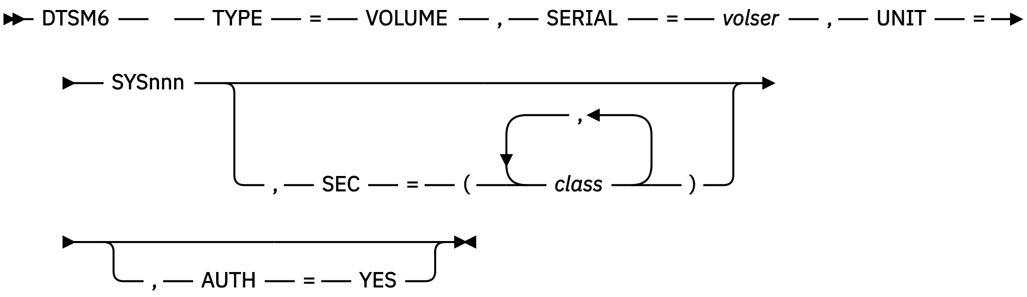


The maximum value for n is 3600. The default if this table entry is not made is 400. One unit of space is:

- One track for a CKD device
- 16 blocks for an FBA device.

Define Volumes for VSE/ICCF Use

Any disk volume on which terminal users may define normal VSE files (as opposed to dynamically allocated files) via the /FILE job entry statement, should be specified by an entry in this table. In addition, if only authorized users or users of a certain security level are able to access the volume, this also should be specified. The format of the table entry is:

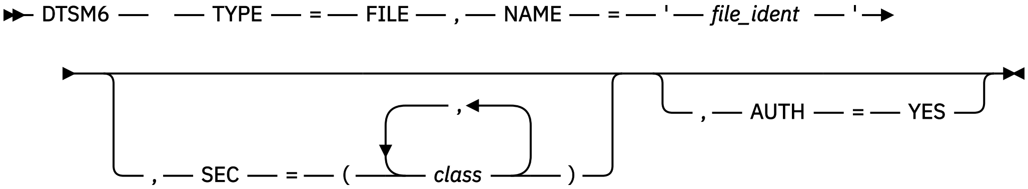


where volser = the volume serial number
SYSnnn = the SYS number assigned to the volume in the VSE/ICCF
initialization deck.

The SEC= and AUTH= operands are optional. If SEC=(1,4,6) is specified, for example, only users with security class (or levels) 1, 4 or 6 can define files on the volume. If AUTH=YES is specified, only authorized users can access the volume. The default is AUTH=NO. If AUTH=YES is specified, your SEC= specification is ignored. Volumes on which dynamic disk space areas are defined need not be specified in this table unless the volume also contains non-dynamically allocated files which terminal users will reference during interactive partition executions via the /FILE statement. If there are users at your installation who are allowed to dynamically allocate permanent files (DISP=KEEP), remember that these files are accessible for subsequent executions. Therefore, the volumes on which these files will reside have to be specified in the System File Table.

Define Restricted Files

If users have access to given volumes but there are files on those volumes, which are to be restricted to only authorized users (bit 1 of OPTB in the user profile) or to users of a given security class, those files should be defined in the table as follows:



where 'file_ident' = The file identification as it appears on
disk (in the VTOC).

With AUTH=YES, only authorized users will be able to define the file on a /FILE statement and thus access the file. If SEC=(6,7,8) is specified the file can be defined and accessed by users whose security class is 6, 7, or 8. If AUTH=YES is specified, SEC= is ignored.

Example for Defining an Entire System File Table

Figure 8 on page 52 shows the table as it is delivered with VSE/ICCF. The table resides in the VSE system sublibrary (IJSYSRS.SYSLIB) as source member DTSSYSFL.A. It resides in this sublibrary also as member DTSSYSFL.PHASE.

```
   DTSM6 TYPE=MAXSPC,TRACKS=400  MAX SINGLE ALLOCATION ENTRY
   DTSM6 TYPE=VOLUME,SERIAL=DTSLIB,UNIT=SYS011
   DTSM6 TYPE=VOLUME,SERIAL=111111,UNIT=SYS001
   DTSM6 TYPE=VOLUME,SERIAL=ICFPAK,UNIT=SYS013
   DTSM6 TYPE=FILE,AUTH=YES,NAME='ICCF.FILE'
   DTSM6 TYPE=FILE,NAME='PAY FILE',SEC=(5,6,7)
```

Figure 8. System File Table as Shipped with VSE/ICCF

Modifying the System File Table

To modify the System File Table, run job SYSFL, which is shown in Figure 9 on page 52. You can do either of the following:

- Code all DTSM6 statements, repeating those in the IBM-supplied source book DTSSYSFL.A (possibly with modifications) and adding your own.
- Create a copy of the IBM-supplied source book DTSSYSFL.A and modify or add DTSM6 statements in that copy.

  In the job stream shown in Figure 9 on page 52 your only input would then be a 'COPY name' statement instead of all the individual DTSM6 statements (where 'name' is the name that you chose for your copy).

```
// JOB SYSFL
// LIBDEF PHASE,CATALOG=IJSYSRS.SYSLIB
// OPTION CATAL
   PHASE DTSSYSFL,*
// EXEC ASSEMBLY
         START 0
         DTSM6 TYPE=MAXSPC,...
         DTSM6 TYPE=VOLUME,...
         . . .
         END
/*
// EXEC LNKEDT,PARM='MSHP'
/&
```

Figure 9. Example Job – Modify the System File Table

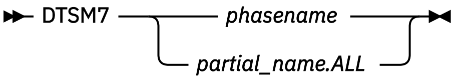
After completion of this job, a new copy of the phase DTSSYSFL.PHASE is cataloged in the VSE system sublibrary. If VSE/ICCF is running at the time the assembly and link edit are performed, the new version of the table will not take effect until the next time VSE/ICCF is started.

Load Protection Table

The Load Protection Table defines phases whose availability to normal terminal users is restricted. Every load or fetch SVC is monitored against this table. It thus provides protection additional to the authorized user facility of the System Program Table, which only checks the first phase loaded in a job step.

Define Load Protection Table Entry

Each entry in the table is defined by one of the following statements:



where:
  phasename = The name under which the restricted phase is
                  cataloged in the VSE system sublibrary.
  partial-name = A partial phase name of one to seven characters.

For example, if IJDD.ALL was specified, any phase name which begins with the characters 'IJDD' would be considered restricted.

Example for Defining an Entire Load Protection Table

This example of a Load Protection Table shows the table as it is delivered with VSE/ICCF. This source book resides under the name DTSSYSLD.A in the VSE system sublibrary. The phase of the table example is cataloged as DTSSYSLD.PHASE in the VSE system sublibrary.

```
  DTSM7 DPQ
  DTSM7 DTSANALS
  DTSM7 ECM.ALL
```

Modify the Load Protection Table

To modify the Load Protection Table, run the job SYSLD, which is shown in Figure 10 on page 53. You can do either of the following:

- Code all DTSM7 statements, repeating those in the IBM-supplied source book DTSSYSLD.A (possibly with modifications) and adding your own.
- Create a copy of the IBM-supplied source book DTSSYSLD.A and modify or add DTSM7 statements in that copy.

In the job stream shown in Figure 10 on page 53 your only input would then be a 'COPY name' statement instead of all the individual DTSM7 statements ('name' being the name that you chose for your copy).

```
// JOB SYSLD
// PAUSE
// LIBDEF PHASE,CATALOG=IJSYSRS.SYSLIB
// OPTION CATAL
   PHASE DTSSYSLD,*
// EXEC ASSEMBLY
         START 0
         DTSM7 DTSANALS
         DTSM7 ....
         . . .
         END
/*
// EXEC LNKEDT,PARM='MSHP'
/&
```

Figure 10. Example Job – Modify Load Protection Table

After completion of this job, a new copy of the phase DTSSYSLD.PHASE is cataloged in the VSE system sublibrary (IJSYSRS.SYSLIB). If VSE/ICCF is running at the time the assembly and link edit are performed, the new version of the table will not take effect until the next time VSE/ICCF is started.

Note: Depending on the number of entries in the above tables, it may be necessary to place the tables in the SVA because they may not fit into the VSE/ICCF control program.

The VSE Access Control Table (DTSECTAB)

If the access control function of the VSE System is active, the above-mentioned access control functions of VSE/ICCF are bypassed. In this case any resource to be protected must be defined in the VSE access control table. The DTSECTAB macro is used to establish the VSE access control table (phase DTSECTAB).

Users accessing protected resources must be defined in the VSE Control File.

For a detailed description of the DTSECTAB macro and how to define users in the VSE Control File refer to z/VSE Administration.

Tailoring the VSE/ICCF Terminal Control Table (DTSTCT)

It may be necessary for you to tailor your VSE/ICCF terminal control table, either prior to setting VSE/ICCF into operation or at any time thereafter. The table resides as a phase in the system sublibrary (IJSYSRS.SYSLIB). It must be cataloged under the name DTSTCT.PHASE and is loaded during the startup of VSE/ICCF.

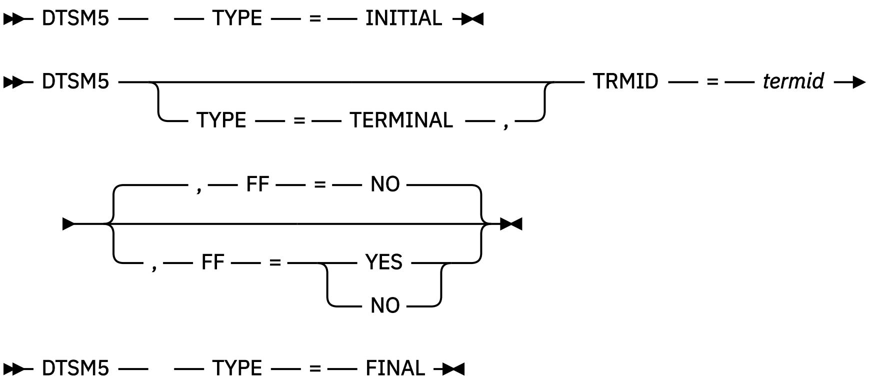
In this table you should define an entry for each hardcopy printer of your installation that accepts form feed control characters and for which you request VSE/ICCF's form feed support. VSE/ICCF will:

- Insert a form feed character at the beginning of every /LISTP, /LOCP, and /SKIP operation if issued during /LISTP.
- Insert a form feed character in place of every skip-to-new-page occurring in a VSE/POWER list file that is listed by /LISTP on the hardcopy printer.

For a hardcopy printer that is not defined in the table, or for which form feed support is not requested in the table, two blank lines will be generated by VSE/ICCF in the cases mentioned above.

Definition of the VSE/ICCF Terminal Control Table

You define your VSE/ICCF terminal control table through the DTSM5 macro which is a source book in the system sublibrary (IJSYSRS.SYSLIB). The format of the macro is as follows:



TYPE=INITIAL
Must be the first DTSM5 macro specified. It must not have any other operands.

TYPE=TERMINAL
Generates an entry in the VSE/ICCF terminal control table for a hardcopy printer for which you request form feed support for VSE/POWER list files.

TRMID=termid

Is a unique four-character identification for the hardcopy printer (identical to the corresponding terminal ID in the CICS terminal control table). This operand is mandatory in a DTSM5 TYPE=TERMINAL macro.

FF=

Specifies whether or not you want VSE/ICCF form feed support for VSE/POWER list files that are printed on the hardcopy printer defined by this DTSM5 TYPE=TERMINAL macro. The default is FF=NO.

TYPE=FINAL

Must be the last DTSM5 macro specified. It must not have any other operands.

Assemble and Catalog the VSE/ICCF Terminal Control Table

Run a job similar to the one shown in Figure 11 on page 55.

```
// JOB TRMASSEM
// LIBDEF PHASE,CATALOG=IJSYSRS.SYSLIB
// OPTION CATAL
   PHASE  DTSTCT,*
// EXEC  ASSEMBLY
         DTSM5 TYPE=INITIAL
         ...                     ONE OR MORE
         ...                     DTSM5 TYPE=TERMINAL
         ...                     MACROS
         DTSM5 TYPE=FINAL
         END
/*
// EXEC LNKEDT,PARM='MSHP'
/&
```

Figure 11. Example Job – Assemble and Catalog the VSE/ICCF Terminal Control Table

Optimizing Performance

The performance of VSE/ICCF generally relates to:

- Individual terminal response.
- Throughput of jobs scheduled in interactive partitions.

The following discussion should help you in maintaining VSE/ICCF so that you achieve a good level of **individual terminal response**.

Terminal response (that is the time lag between when a command or data is entered and when the response appears at the terminal) may be influenced by tailoring options, by user profile specifications, by installation-defined file maintenance procedures and by the way the system is used by terminal users. Below are some specific factors influencing performance.

Tailoring Options

The NBUFS and NRECS/CISIZE options in some ways influence foreground performance in that the more buffer areas there are in storage, the less will be the need to perform physical I/O to handle foreground command requests.

User Profile Considerations

The way the user profile is established will ultimately have an effect on foreground performance.

- The *maximum number of data records* per library member affects foreground performance. This number should be defined as small as possible, consistent with your requirements. A value in the range between 500 and 1000 is recommended. Remember that a given value here does not prevent you from preparing larger programs. Many members may be logically grouped together as a single logical member using the /INCLUDE facility.

- The *size and number of libraries* will have a definite effect on foreground performance. It is much more efficient to have several small libraries rather than a few large libraries. With several small libraries, the directories will be smaller and thus the time to look up a given member will be shorter.
- The total *number of user profiles defined within the system* can have an effect on response especially if users are frequently logging on and off. It would be good practice to restrict the number of user profile records to less than 300.
- It is more efficient to have *often used user profile records towards the front* rather than towards the end of the set of profile records.

Library Considerations

When defining a library, specify a maximum number of directory entries. This limits the size of the directory and forces a user to purge members that are no longer required. This in turn helps to reduce the total space requirements for the VSE/ICCF library file because purged member records are returned to the file's free chain. IBM recommends an initial value below 200.

Use the FREESPACE option (of the DTSUTIL command ADD LIBRARY) for highly active libraries. If a library has been established with the FREESPACE option, a certain number of free directory entries will be allocated within the directory area each time the library is restored (DTSUTIL RESTORE function). This gives the advantage of concentrating all directory entries, even the ones added online after a restore, into contiguous disk records and blocks, thus reducing the number of physical I/O requests required to scan a directory.

Reorganize the libraries at regular time intervals to maintain an acceptable level of performance. Use the BACKUP/RESTORE functions of DTSUTIL. The RESTORE function of DTSUTIL for the complete library file will organize all directory entries into contiguous records and blocks, which reduces the number of physical I/O operations required to perform a directory look-up. The BACKUP/RESTORE function places all logically related records into physically contiguous storage locations and compresses the file. All of these activities enhance terminal response time.

Do not use selective restore for improving library performance because this would result in performance degradation as described in Note 1 of section "RESTORE (Library) Command" on page 142.

VSE/ICCF initialization message K088I describes the number and percentage of high file records. When this number reaches zero, a noticeable performance degradation may occur because available records for additions to members, spool areas, log areas, and directory records will be scattered into more physical records. It is a good idea to notice the rate of change that this number of available records undergoes and to do a BACKUP/RESTORE before it reaches zero.

Miscellaneous Considerations

One of the easiest ways of improving terminal response is something that the terminal user can influence. This involves use of multiple line or command input in a single terminal transmission. The user separates each command or data line with the line end character. If the user does not have to wait for a terminal response between each line or command entered, the amount of work done in a given period of time should increase.

You can encourage the user to employ this technique by defining a line end character in the user's profile record or in a logon procedure (see "Adding or Changing a User Profile" on page 128).

In addition, the /SET DELAY command may be issued to completely eliminate all output from commands other than the last command or line of the multi-line input. This should also reduce response time.

The user of an IBM 3270 Display System may issue the /SET SCREEN command to increase the size of the input area from 1 line to 2, 3 or 4 lines (up to a maximum of 255 characters), thus allowing even more commands or data lines to be entered in a single terminal transmission.

Storage Requirements

All VSE/ICCF components, except the CICS related transaction modules of VSE/ICCF (DTSICCF, DTSPOSTI and DTSSHUT) are located within the CICS GETVIS area below 16MB.

The following values reflect the z/VSE system as delivered to you by IBM. This system assumes

5 interactive partitions, standard layout
NBUFS=20
CISIZE=2048 (for FBA) or NRECS=22 (for CKD)

```
VSE/ICCF control part during initialize              236   K   see Note 1
Interactive partition control blocks                   6   K   see Note 2
DTSFILE buffer area                                   43   K   see Note 3
Miscellaneous, approximately:                          1   K
------------------------------------------------------------
Entire GETVIS space during initialize                286   K

Interactive partition 1                             1024   K
Interactive partition 2                              384   K
Interactive partition 3                              384   K
Interactive partition 4                              512   K
Interactive partition 5                              512   K
------------------------------------------------------------
Entire GETVIS space during initialize, with
all standard interactive partitions allocated       3102   K

Each user logged on to VSE/ICCF, approximately:        1.6 K

PFIXed storage, VSE/ICCF control                      16   K
PFIXed storage, control blocks                         6   K  see Note 4
```

Note:

1. After initialize is completed, the following amount of GETVIS storage is free from the end of this area 18 K
2. For each additional interact. partition, add: 1 K
3. For each buffer in NBUFS above 20, add: .3 K
4. Calculation of file buffers

   For FBA devices: CISIZE * NBUFS

   For CKD devices: NRECS * NBUFS
5. For each additional interactive partition, add PFIXED storage: 1 K

Each area by itself requires contiguous storage.

Example Calculation

This calculation assumes a standard VSE/ICCF system with 6 users logged on to VSE/ICCF.

```
Control part (minus 18K)                             218   K
Interactive partition control blocks                   6   K
DTSFILE buffer area                                   43   K
Miscellaneous, approximately:                          1   K
Interactive partition 1                             1024   K
Interactive partition 2                              384   K
Interactive partition 3                              384   K
Interactive partition 4                              512   K
Interactive partition 5                              512   K
6 users logged on to VSE/ICCF, approximately:         10   K
------------------------------------------------------------
Entire GETVIS space requirement for VSE/ICCF        3094   K

For each VSE/ICCF full screen editor
session, add approximately:                            1   K
------------------------------------------------------------

PFIXed storage regardless of number
of logged-on users                                    24   K
------------------------------------------------------------
```

```
Space requirement for VSE/ICCF's
SVA eligible phases:
    LINKNGO2                                           7   K
    DTSCDUMP                                           9   K
    DTSPROCS                                          10   K
    DTSSBMT1                                          10.5 K
    DTSXTRCT                                            .5 K
    DTSIPWR                                            4   K
____________________________________________________________
Entire SVA storage requirement, approximately:        41   K

VSE/ICCF transaction program requirement:
    DTSICCF                                           18   K
    DTSPOSTI                                           4.5 K
    DTSSHUT                                             .5 K
____________________________________________________________
Entire transaction program requirement                23   K
```

Chapter 3. Operation

There are two important rules for creating and maintaining a successfully operating VSE/ICCF environment:

1. If a system failure occurs and VSE/ICCF cannot go through its normal or abnormal termination routine, then the RECOVER function of the DTSANALS utility must be run (see "16. Abnormal End Recovery" on page 63).
2. The VSE/ICCF library file should be backed up and restored at least once a month. The recommended interval is one week. For an example set of required control statements, see "1. Reorganizing the VSE/ICCF Library File" on page 60.

Initialization and Termination Choices

VSE/ICCF can only be used in a (static or dynamic) partition with an active CICS. VSE/ICCF must reside below the 16MB storage boundary.

Initializing VSE/ICCF

VSE/ICCF is initialized automatically during the CICS *post-initialization phase* or after you enter the transaction I$ST.

PLTPI (Program List Table Post-Initialization) Programs

In z/VSE, the VSE/ICCF initialization phase DTSPOSTI is specified in the PLTPI table. Therefore VSE/ICCF is initialized whenever CICS is started under control of the PLTPI table.

Starting VSE/ICCF and CICS together ensures that **contiguous GETVIS storage** required for VSE/ICCF is available.

Transaction I$ST [phasename]

Use this transaction to initialize VSE/ICCF under any of the following conditions:

- You did not specify DTSPOSTI in the PLTPI table.
- You had terminated VSE/ICCF and want to restart it.
- You want to start a VSE/ICCF generation phase that is different from the default phase DTSIGEN.

The optional operand *phasename* specifies a cataloged VSE/ICCF generation phase (see Figure 2 on page 32). If you do not specify a phase name, the default phase DTSIGEN is used.

The VSE/ICCF utilities always take their information from DTSIGEN.

All VSE/ICCF initialization messages are routed to the system console regardless from where you started the transaction: from a terminal or from the system console. In either case, decision-type messages must be answered at the system console.

Terminating VSE/ICCF

VSE/ICCF can be terminated in three ways:

- as part of the CICS shutdown
- by entering a VSE/ICCF operator command
- by entering a transaction name.

PLTSD (Program List Table Shutdown)

The phase DTSSHUT is specified in this table. During the first quiesce stage of the CICS shutdown, this phase is called and performs a controlled shutdown of VSE/ICCF.

Operator Command /ICCFEND

Please refer to the detailed description in section "/ICCFEND Command" on page 73.

Transaction I$SH

This transaction shuts down VSE/ICCF regardless of any pending VSE/ICCF transactions. It is generally recommended to use the /ICCFEND command, which waits for the transactions to complete and terminates VSE/ICCF in a more controlled manner. However, if you do have to use the transaction I$SH, make sure to force or purge any pending ICCF transactions after I$SH has completed.

If transaction I$SH was initiated from a terminal, this terminal will be unlocked when the transaction has terminated. Any related messages will be written to the system console.

This transaction should be used only in exceptional cases.

Immediate Shutdown of CICS

During an immediate shutdown of CICS, the PLTSD phases do not get control. Instead, VSE/ICCF will be terminated through its own abend routines.

This might cause the update-in-progress (UPIP) attribute to remain on for members that are currently in edit mode. The VSE/ICCF administrator should use the /PROTECT command to clear up this situation. Refer to " /PROTECT" in the VSE/ICCF User's Guide.

Initialization Job Stream

This section describes a general initialization job stream. It is rather comprehensive in that it touches on all aspects of such a job stream. At the end of the section, under "Skeleton for Starting CICS Transaction Server with VSE/ICCF (SKCICS)" on page 68, the job to start CICS with VSE/ICCF delivered withz/VSE is shown. This job stream is specifically geared toward the z/VSE environment.

Generally speaking, once the initialization procedure has been set up it may be used for initialization in a day-to-day environment. This procedure is very flexible and can easily be changed to meet new requirements, for example when you want to add new functions or alter performance characteristics.

First, let us look at an outline of the JCL recommended or required to initialize VSE/ICCF. The numbered headings refer to detailed explanations that follow this job stream.

Note: In addition to the devices used during VSE/ICCF operation, dummy devices may have to be added during IPL. Refer to the description of the RDR operand in "Description of VSE/ICCF Tailoring (DTSOPTNS) Options" on page 24.

Outline of Initialization JCL

Some of the jobs contain a call of procedure DTRICCF. This procedure provides the ASSGN for the VSE/ICCF library file DTSFILE. DLBL/EXTENT information is not shown because it is assumed to be stored in the standard label information area.

1. Reorganizing the VSE/ICCF Library File

```
// JOB REORG LIBRARY AND CREATE BACKUP
// PAUSE TYPE CANCEL TO BYPASS BACKUP/RESTORE
// ASSGN SYS005,TAPE           (Tape Output)
// TLBL DTSBKUP,'DTSBKUP'
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSUTIL
BACKUP
```

```
/*
// MTC REW,SYS005
/&
// JOB RESTORE
// PAUSE CHECK THE LIST OUTPUT OF THE PRECEDING BACKUP JOB
// ASSGN SYS005,UA
// ASSGN SYS004,TAPE           (Tape Input)
// TLBL DTSRSTR,'DTSBKUP'
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSUTIL
RESTORE
/*
/&
```

2. Adding Mail or Broadcast Records

Adding Mail:

```
* MAIL IS KEPT AS A MEMBER IN THE
* COMMON LIBRARY (LIB 2 HERE)
// EXEC DTSUTIL
PURGE LIB 2 MEMBER(A$MAIL)
ADD MEMBER(2,A$MAIL,AAAA)
 ...
 ...                            general and user mail
 ...
END OF MEMBER
/*
```

Adding Broadcast Records:

```
// JOB ADD BROADCAST RECORDS
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSUTIL
ADD BROADCAST
* -----                        broadcast information must have
* -----                        an * in the first column
  ...                          (up to 8 records)
END
/*
/&
```

3. Initialization Job Control

```
* $$ JOB JNM=ICCF,CLASS=2,DISP=L
* $$ LST DISP=D,CLASS=A,RBS=100
// JOB START  CICS Transaction Server WITH VSE/ICCF
```

4. UPSI Settings

```
// UPSI abcdefgh                CICS Transaction Server UPSI settings
```

5. Assignments

```
// EXEC PROC=DTRICCF            assign DTSFILE

( procedure DTRICCF is assumed to have the following content:
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR
// ASSGN SYS011,DISK,VOL=SYSWKn,SHR. )
```

6. Work File Assignments

```
// ASSGN SYS000,SYS010          compiler work file units
// ASSGN SYS001,SYS010
// ASSGN SYS002,SYS010
// ASSGN SYS003,SYS010
// ASSGN SYS004,SYS010
// ASSGN SYS012,DISK,VOL=SYSWKn,SHR   special work file
// ASSGN SYS014,DISK,VOL=SYSWKn,SHR   user permanent file
```

7. Programmer Logical Units

```
// ASSGN  SYS005,UA             unit record devices
// ASSGN  SYS006,UA
// ASSGN  SYS007,UA
// ASSGN  SYS008,UA
// ASSGN  SYS009,SYSLOG
```

8. Tape Assignment

```
// ASSGN  SYS029,280            tape unit – if desired
```

9. Label Information for Pre-Allocated Work Files

The following set of label definitions is taken from the procedure STDLABUP.PROC in the sublibrary IJSYSRS.SYSLIB. Label statements only for the first two interactive partitions are shown. Procedure STDLABUP.PROC actually has label statements for the first **four** interactive partitions.

As shown below, the work files are defined as VSAM files. You can, however, define them also as non-VSAM sequential files.

```
// DLBL IKSYS11,'%WORK.FILE.N11',0,VSAM,                                X
             CAT=IJSYSCT,RECSIZE=4096,                                  X
             DISP=(NEW,DELETE),RECORDS=(20,20)
// DLBL IKSYS12,'%WORK.FILE.N12',0,VSAM,                                X
             CAT=IJSYSCT,RECSIZE=4096,                                  X
             DISP=(NEW,DELETE),RECORDS=(20,20)
// DLBL IKSYS13,'%WORK.FILE.N13',0,VSAM,                                X
             CAT=IJSYSCT,RECSIZE=4096,                                  X
             DISP=(NEW,DELETE),RECORDS=(20,20)
// DLBL IKSYS14,'%WORK.FILE.N14',0,VSAM,                                X
             CAT=IJSYSCT,RECSIZE=4096,                                  X
             DISP=(NEW,DELETE),RECORDS=(20,20)

// DLBL IKSYS21,'%WORK.FILE.N21',0,VSAM,                                X
             CAT=IJSYSCT,RECSIZE=4096,                                  X
             DISP=(NEW,DELETE),RECORDS=(20,20)
// DLBL IKSYS22,'%WORK.FILE.N22',0,VSAM,                                X
             CAT=IJSYSCT,RECSIZE=4096,                                  X
             DISP=(NEW,DELETE),RECORDS=(20,20)
// DLBL IKSYS23,'%WORK.FILE.N23',0,VSAM,                                X
             CAT=IJSYSCT,RECSIZE=4096,                                  X
             DISP=(NEW,DELETE),RECORDS=(20,20)
// DLBL IKSYS24,'%WORK.FILE.N24',0,VSAM,                                X
             CAT=IJSYSCT,RECSIZE=4096,                                  X
             DISP=(NEW,DELETE),RECORDS=(20,20)
```

10. Special Work File

```
// DLBL   IKSYS15,,0
// EXTENT SYS012,SYSWKn,1,0,mm,nn
```

11. Redefine Work File

```
// DLBL   IKSYS19,,0,DA             special work file
// EXTENT SYS000,SYSWKn,1,0,mm,nn
```

12. Define Dynamic Space Areas

```
// DLBL   DTSDYNC,,0,DA
// EXTENT SYS001,SYSWK1,1,0,mm1,nn1
// EXTENT SYS001,SYSWK1,1,1,mm2,nn2
// DLBL   DTSDYNW,,0,DA
// EXTENT SYS003,SYSWK1,1,0,mm3,nn3
```

13. Define the VSE/ICCF Library File and VSE Sublibraries

```
// DLBL   DTSFILE,'ICCF.LIBRARY',1999/365,DA
// EXTENT SYS010,SYSWK1,1,0,mm,nn
// EXTENT SYS011,SYSWKn,1,1,mm,nn

// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASED,PRD1.BASE,PRD2.PROD,         X
               PRD2.CICSR,PRD2.DBASE,PRD1.MACLIBD,PRD1.MACLIB),PERM
```

14. Define User Permanent Files

```
// DLBL   SCHEDFL,'STUDENT REG FILE',99/365
// EXTENT SYS014,SYSWKn,1,0,mm1,nn1
// DLBL   ACCTFL,'ACCOUNT FILE',99/365,DA
// EXTENT SYS014,SYSWKn,1,0,mm2,nn2
```

15. Start CICS Transaction Server with VSE/ICCF

```
// SETPFIX LIMIT=144K
// EXEC DFHSIP,SIZE=...
*  VSE/ICCF WILL BE STARTED VIA CICS POSTINITIALIZATION PHASE

 . . .
 . . .
* $$ EOJ
```

16. Abnormal End Recovery

```
// JOB RECOVER FROM ABNORMAL TERMINATION
*  (DLBL/EXTENT AND ASSGN FOR DTSFILE, IJSYS01, AND IJSYS02
*   IF REQUIRED)
// EXEC DTSANALS
RECOVER
/*
/&
```

Discussion of the Initialization JCL

The preceding JCL outlines the possible parts of a procedure for starting VSE/ICCF. These parts (numbered accordingly) are discussed below.

1. Reorganizing the VSE/ICCF Library File

You should perform a BACKUP and RESTORE function at least once a month, better yet once every week. This provides two significant benefits. First, a backup file is available in case system or hardware problems cause the library file to become unusable, or if a portion of the file is overwritten by some other program. Secondly, the BACKUP/RESTORE operation causes the file to be reorganized which means that all associated records are assigned to contiguous locations. Directory lookup, acquiring of free records and sequential retrieval all become faster. Note that by using the DATAONLY option of the RESTORE command prevents accumulated accounting information from being lost in the event the RESTORE does not immediately follow the BACKUP.

VSE/ICCF initialization message K088I describes the number and percentage of high file records. When this number reaches zero, a noticeable performance degradation may occur because available records for additions to members, spool areas, log areas, and directory records will be scattered into more physical records. It is a good idea to notice the rate of change that this number of available records undergoes and to do a BACKUP/RESTORE before it reaches zero.

2. Adding Mail or Broadcast Records

The installation may choose to add up to eight broadcast records to the VSE/ICCF library file prior to starting the system. The broadcast record(s) are displayed to each user during logon. These records must start with an asterisk character and may contain any descriptive information.

The mail member (A$MAIL) should be placed in the common library (library 2 in the example shown in "2. Adding Mail or Broadcast Records" on page 61). If the entire mail member is not to be replaced, the DTSBATCH utility may be used to edit the existing mail member.

Note:

1. The member A$MAIL may also be updated or replaced **interactively** by the VSE/ICCF administrator.
2. For the format of A$MAIL records refer to "Mail Format" on page 80.

3. Initialization Job Control

This is the beginning of the JCL for actually starting CICS with VSE/ICCF. When running under VSE/POWER, the VSE/POWER job statement should indicate a class which will cause the proper VSE partition to be entered.

You should include the VSE/POWER LST statement with RBS=100 to utilize VSE/POWER's segmentation option. This is useful when you issue the VSE/ICCF operator command /FDUMP.

4. UPSI Settings

The UPSI bits are used by CICS only.

The VSE/ICCF options which in previous releases could be selected by UPSI bits are now VSE/ICCF tailoring options. Refer to the description of the tailoring options DYNSPC, FILEVER and KATAKAN of the DTSOPTNS macro in "Description of VSE/ICCF Tailoring (DTSOPTNS) Options" on page 24.

5. Assignments

The ASSGN statements for the VSE/ICCF library file are stored in procedure DTRICCF. In this documentation, procedure DTRICCF is assumed to have the following content:

```
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR
// ASSGN SYS011,DISK,VOL=SYSWKn,SHR
```

At least one SYS number should be assigned to each disk accessible through VSE/ICCF interactive partition executions. This means that each disk which has pre-allocated work areas, dynamic space areas, the library file itself or any other file which will be opened in an interactive partition must have a SYS number assigned to it.

With respect to the VSE/ICCF library file, it is recommended to assign SYS010 to the disk drive with the first extent, and (as required) SYS011 to the second, SYS012 to the third, etc. However, the user may assign any SYS numbers which do not conflict with the unit record devices that are defined via the PGMRLST, PGMRINP, PGMRPUN, PGMRPIN, and PGMRLOG tailoring operands. The defaults for these units are SYS005, 6, 7, 8, and 9.

6. Work File Assignments

You may ignore this discussion on work file assignments if your pre-allocated work files reside in VSAM-managed space (which is the case in z/VSE).

The SYS numbers SYS000, 1, 2, 3 and 4 must always be assigned and they must be assigned to the same device type to allow compilers to function properly. To be consistent with standard VSE usage, you may assign SYS000, 1, 2, 3 and 4 to the devices where the pre-allocated work files are defined and use these SYS numbers on the // EXTENT statements for the files. If instead of pre-allocated work files dynamic space is used, SYS000, 1, 2, 3 and 4 must be assigned to those devices on which the dynamic space area has been defined.

It is, however, possible to use any appropriately assigned SYS number (for example SYS011 or SYS012) when filling out the // EXTENT specifications for the pre-allocated work files as long as SYS000, 1, 2, 3 and 4 are assigned to some disk device(s) and the device(s) is(are) of the same type as the assignment of the SYS number in the // EXTENT card.

Note: There may be certain programs which depend on the fact that file IJSYS0n is on the drive assigned to SYS00n. In these instances, the appropriate SYS number should be used in the // EXTENT, and all versions of the file name (for example, IKSYS11, IKSYS21, IKSYS31 are all versions of IJSYS01, for interactive partitions 1, 2 and 3) must be on the same drive. If it is not possible or desirable to have all versions of the work file (that is, IJSYS01) on one drive, use the /SET CLASS command to select the interactive partition for which the work file meets your program's requirements.

7. Programmer Logical Units

The SYS units which the user specified as PGMRINP, PGMRLST and PGMRPUN (the defaults are SYS005, SYS006 and SYS007) during the tailoring of VSE/ICCF should be assigned to UA in partitions where there are no physical unit record devices. In partitions with unit record devices (partitions running under control of VSE/POWER are considered to have unit record devices), these units may be (but need not be) assigned to SYSIPT, SYSLST and SYSPCH (or to UA). (When the SYS units are assigned to UA, VSE/ICCF will internally assign the units to the generated dummy devices if SYSIPT, SYSLST, SYSPCH are not assigned for the partition; otherwise the assignments are used.)

The SYS unit specified as PGMRPIN (the default is SYS008) during tailoring of VSE/ICCF must be assigned to UA.

The SYS unit specified as PGMRLOG (the default is SYS009) during tailoring of VSE/ICCF must be assigned to SYSLOG.

8. Tape Assignment

To process tape files from interactive partitions, the tape drive must be allocated to the CICS/ICCF partition during the entire time VSE/ICCF is running. Furthermore, tape processing is not protected among different interactive partitions. For example, one interactive partition may have positioned the tape to a certain file and is now relying on this positioning but in the meantime another interactive partition could do a rewind on that same tape thus having the tape incorrectly positioned for the first interactive partition.

9. Label Information for Pre-Allocated Work Files

If all compiler and temporary work file requirements will be allocated from the dynamic space areas, this reference number can be be ignored.

Pre-allocating work files for some or all interactive partitions brings some performance advantage due to file placement. It is possible to have pre-allocated work files for certain often used interactive partitions and to have work files dynamically allocated in others. Whether or not a particular interactive partition has pre-allocated work files should be indicated in the tailoring options PARTN and PARTX.

The rest of the description under this reference number applies to defining pre-allocated work files for those interactive partitions which will have this type of support.

If a job were to execute in interactive partition 1 and were to reference the file named IJSYS02, for example, this would be converted by VSE/ICCF into a reference to IKSYS12. If the same job were run in interactive partition 4, the reference to IJSYS02 would become a reference to IKSYS42.

A set of work files (equivalents of IJSYS01, 2, 3, up to 4 depending on the 'y' specification in the tailoring option PARTN) must be defined for each interactive partition within the system which is to have pre-allocated work files. The J in the file name operand (IJSYS0n) of the // DLBL statement must be changed to K and the second to the last character must be changed from a 0 to the interactive partition identifier (1 through 9 and A through Z).

It is also possible to specify more than the required four temporary work files. For example, for work file IJSYS00 specify IKSYS10, IKSYS20, etc., for IJSYS05 specify IKSYS15, IKSYS25, etc. If the work files IJSYS05, 6, 7, 8, 9 are defined, do not use SYS005, 6, 7, 8, 9 in an EXTENT statement. Instead, use the

VSE/ICCF disk SYS number (SYS010 through SYS01n) which corresponds to the drive on which the file resides.

10. Special Work File

In this example, a sixth work file (IJSYS05 - that is, IKSYS15) is defined for interactive partition 1.

11. Redefine Work File

In this example, the work file IJSYS09 is defined for interactive partition 1 (IKSYS19); however, it is defined as a direct access (DA) file which redefines the IJSYS00 area. This might be useful if the installation wished to provide its users a temporary area for testing direct access file type programs.

A sequential file could be copied into the IJSYS00 area and then referenced in a random fashion by calling the file IJSYS09.

12. Define Dynamic Space Areas

If you do not intend to make use of the VSE/ICCF dynamic disk space allocation facility, ignore the JCL at reference 12.

Disk space can be dynamically allocated for jobs in interactive partitions if dynamic disk space is defined. This disk space may consist of up to 9 extents. If two or more such extents are defined, they all must reside on disks of the same type. All extents on a volume must have the same logical unit (SYSnnn) assignment.

An extent cannot have more than 7,200 tracks for count-key-data disks or 115,200 blocks for FBA disks.

You can define the dynamic space as one or two direct access files with the file names DTSDYNC or DTSDYNW. Use the file name:

- DTSDYNC if the extents of the file are to be cold started, unless the operator explicitly requests a warm start.
- DTSDYNW if the extents of the file are to be warm started.

During its startup, VSE/ICCF refers to the dynamic space area with the file name DTSDYND. You might come accross this name in an initialization message.

If you do not supply label information for the files DTSDYNC and DTSDYNW, no disk space is available for dynamic allocation.

Note: After startup, the disk space you defined for dynamic allocation is not protected. There is no file label for this space in the VTOC of the containing volume. A listing of the VTOC shows this area to be free, except for the portions occupied by currently open and dynamically allocated files.

For reasons of data integrity, ensure that no other file is defined
(by DLBL and EXTENT statements) within your dynamic space areas.

In an interactive partition you can make use of dynamic disk space allocation for access methods as follows:

- SAM and DAM if the space is defined on a CKD device.
- FBA-SAM if the space is defined on an FBA device.

At reference number 12, three dynamic space areas (extents) are defined.

Note:

1. When using different device types for the dynamic space area, think of your compiler work files. They require that the device type of SYS001 through SYS004 agrees with the device type actually used for these files.
2. You determine via your /FILE statement from which dynamic space area your dynamic space segment is satisfied. The /FILE operands controlling the space allocation are: NAME, IDENT, SERIAL, UNIT and VOLUME. Refer to " /FILE" in the VSE/ICCF User's Guide.

3. Refer to the documentation of message K479D in z/VSE Messages and Codes 1 for possible initialization options of dynamic space.
4. A certain amount of space (at least one track) will be used by VSE/ICCF internally to manage the dynamic space area. This area contains control records and is therefore not available for allocation.

13. Define the VSE/ICCF Library File and VSE Sublibraries

Normally, the DLBL/EXTENT statement set for the VSE/ICCF library file (DTSFILE) should be stored in the VSE standard label area. However, it is shown here for completeness. Notice that the file occupies two extents on two volumes. Please be aware that for DA-type files the SYS numbers must be in ascending order.

In the CICS/ICCF partition, define the VSE sublibraries that contain VSE/POWER, CICS Transaction Server, VSE/ICCF, and any other library that needs to be accessed in order to run VSE/ICCF. For this purpose, your z/VSE system has the following sublibraries defined:

```
PRD2.CONFIG
PRD1.BASED
PRD1.BASE
PRD2.PROD
PRD2.CICSR
PRD2.DBASE
PRD1.MACLIBD
PRD1.MACLIB
```

14. Define User Permanent Files

Any permanent files which are to be used by terminal users running interactive partition jobs can be made available to the user either by providing the label information (DLBL/EXTENT) in the initialization JCL, or in the standard label area, or by defining these files in the job streams using the /FILE statement. However, any multiple extent file to which a user will have access must be defined in the startup job stream. Also, placing a DLBL/EXTENT for a file in the startup job stream means that users may access the file without having to define the file (via the /FILE statement) in their own interactive partition job streams.

15. Start CICS Transaction Server with VSE/ICCF

Finally, the initial phase of VSE/ICCF DTSPOSTI must be invoked. DTSPOSTI is initialized as CICS PLT program in the CICS System Initialization Program DFHSIP.

The allocation for the CICS/ICCF partition must provide sufficient GETVIS storage for VSE/ICCF. The standard VSE/ICCF as shipped by IBM requires approximately 2.5MB.

z/VSE as shipped to you sets a limit of 144K to the CICS/ICCF partition for PFIXing pages. Of this, approximately 24K are adequate for VSE/ICCF's use in most cases.

You may switch VSE/ICCF between two or more CICS Transaction Server systems. Please refer to "Considerations for Switching VSE/ICCF between CICS Systems" on page 69 for information on how to set up and start the other CICS Transaction Server.

Reconfiguration and Terminal Control Initialization:

Reconfiguration of VSE/ICCF at initialization time (which was possible in earlier releases of VSE/ICCF) is no longer supported.

To reconfigure VSE/ICCF you must run a generation job as shown in Figure 2 on page 32. See also the description of transaction I$ST under "Transaction I$ST [phasename]" on page 59.

16. Abnormal End Recovery

Whenever VSE/ICCF ends abnormally such that the message

```
K017I ICCF ABEND PROCESSING COMPLETE
```

does not appear, you must run the RECOVER or REORG function of DTSANALS. Types of termination which warrant running the RECOVER function are:

- System power failure.
- Failure to terminate VSE/ICCF before turning machine power off.
- Processor machine check requiring IPL.
- Termination of a VSE virtual machine under VM without prior VSE/ICCF termination.
- Any other failure which prevents either the normal EOJ or ABEND processing routines to be executed.

If a normal ABEND has occurred (message K017I), it is advisable to execute either the RECOVER function of DTSANALS or the BACKUP/RESTORE function of DTSUTIL. After the ABEND, the disk records in the library file used for $$PRINT, $$PUNCH, and the input areas at the time of the ABEND will be wasted until either DTSANALS RECOVER or REORG or a BACKUP/RESTORE is performed. However, the member records in the library are intact and processing may be restarted with only a loss of some space within the library file.

Skeleton for Starting CICS Transaction Server with VSE/ICCF (SKCICS)

z/VSE already has a particular job for starting CICS with VSE/ICCF. Figure 12 on page 69 shows the startup job.

Skeletons SKCICS and SKLOAD in library 59 assist you in creating your own startup job.

```
* $$ JOB JNM=CICSICCF,DISP=L,CLASS=2,EOJMSG=YES
* $$ 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.SCEECICD,PRD2.SCEECICS,PRD2.SCEEBASD,               X
               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=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF2=ACTIVE'
/*
// EXEC PROC=CPUVAR&XNCPU,XMODEF2,XAPPLF2,XSPINIT,XENVNR
// 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=IJBDUF2,PWD=??????
NOLOG
// EXEC DTSANALS                       RECOVER IF DTSFILE DESTROYED
RECOVER OPT
/*
*   WAITING FOR VTAM TO COME UP
// EXEC IESWAITT
/*
// SETPARM ELIM=10M
// IF XSPINIT = INSTALL THEN
// GOTO NOSEC
// IF XENVNR = B THEN
// SETPARM ELIM=25M
// SETPARM CICSEC=YES
// GOTO SETMODE
/. NOSEC
// SETPARM CICSEC=NO
/. SETMODE
// IF XMODEF2 = COLD THEN
// GOTO STARTCIC
// SETPARM XMODEF2=AUTO
/. STARTCIC
// EXEC DFHSIP,SIZE=DFHSIP,PARM='APPLID=&XAPPLF2,START=&XMODEF2,SEC=&CI*
               CSEC,EDSALIM=&ELIM,SI',DSPACE=2M,OS390
SIT=SP,STATRCD=OFF,XCMD=NO,XDCT=NO,XFCT=NO,XJCT=NO,SVA=NO
XPCT=NO,XPPT=NO,XPSB=NO,XTST=NO,MXT=20
/*
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF2=INACTIVE'
/*
/&
* $$ EOJ
```

Figure 12. IBM-Supplied Job for Starting CICS Transaction Server with VSE/ICCF

Considerations for Switching VSE/ICCF between CICS Systems

You can switch VSE/ICCF between two or more CICS Transaction Server systems.

Each CICS that supports VSE/ICCF must have available all VSE/ICCF-related resources.

VSE/ICCF can be switched between two CICS systems in the following manner. At the system console, enter

```
/ICCFEND        (shut down VSE/ICCF in the partition where it
                is currently running; wait till VSE/ICCF stops running)
MSG xx          (request communication with the CICS subsystem where you
                  intend to restart VSE/ICCF)
nn I$ST          (start VSE/ICCF in partition xx.  nn represents the repl ID)
```

Console-Operator Controls

The console operator communicates with VSE/ICCF via the (VSE/ICCF) operator commands listed below. These commands allow to carry out operations such as displaying active jobs and active users, canceling jobs in interactive partitions, warning users before shutting down the system, and terminating the system. The commands are:

```
/CANCEL             /MAP
/CLASS              /PDUMP
/CONNECT            /SEND
/DISCONN            /TIME
/DISPLAY            /USERS
/FDUMP              /WARN
/ICCFEND
```

VSE/ICCF issues messages whose number have the form **K*nnnx***. For explanations of these messages, refer to z/VSE Messages and Codes 1

Master versus User Console Authority

Depending on how your user profile is defined, your terminal acts as a z/VSE master console or as a z/VSE user console. The z/VSE system console is always a master console.

The master console receives the entire message traffic, and its operator has unrestricted authority to issue any command (z/VSE, VSE/ICCF and others). The user console, on the other hand, receives only those messages that belong to it, such as messages specifically related to a command that was issued from this console.

A user who is defined in the z/VSE Interactive Interface as *Type 1* user or as *Type 2 - OPER* user logs on at a master console.

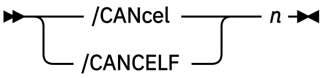
A user who is defined in the z/VSE Interactive Interface as *Type 2 - PROG* user logs on at a user console.

The commands /DISPLAY, /MAP and /USERS can be issued from a master console or from a user console. All other commands in the above list can only be issued from a master console.

VSE/ICCF Operator Commands

The following VSE/ICCF operator commands can all be abbreviated as shown in the command descriptions.

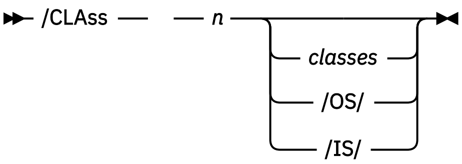
/CANCEL Command



This command causes the job running in an interactive partition to be canceled. n is the partition number and must be an integer from 1 to 35. Use the /DISPLAY command to obtain the identification numbers, current users and status of the various interactive partitions. If the partition appears to accept the CANCEL command but does not cancel, issue the /CANCELF command for the partition. This form of the command should only be issued in situations where the /CANCEL command fails. Otherwise, you could inadvertently terminate VSE/ICCF.

Do not issue /CANCEL nor /CANCELF to cancel an interactive partition after message K754I appeared.

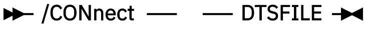
/CLASS Command



This command allows you to alter the scheduling classes (classes operand) associated with a given interactive partition (n operand). The interactive partition is indicated as a decimal number from 1 to 35.

The classes operand may be from one to four alphabetic characters indicating the class or classes to be associated with an interactive partition. Setting a class places an out-of-service partition into service with the new class(es). If /OS/ is specified, the interactive partition is placed out of service with a class setting of zero and therefore may not be scheduled for execution. Specifying /IS/ is the same as specifying class A. If the classes operand is omitted, the current classes associated with the interactive partition will be displayed.

/CONNECT Command



This command reconnects the VSE/ICCF library file which had been disconnected either via the /DISCONN command or by the VSE/ICCF control program during startup.

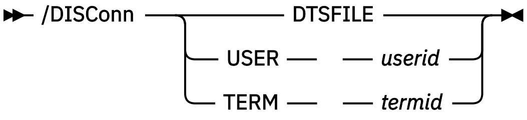
When message

```
K129I DTSFILE IS CONNECTED
```

appears, processing of the /CONNECT command is completed.

Terminal users can now log on to VSE/ICCF again.

/DISCONN Command



DTSFILE
Disconnects the VSE/ICCF library file. This causes all interactive partitions currently active to be canceled (with VSE/ICCF cancel code 15 indicating that the job time limit has been exceeded) and all VSE/ICCF terminals to be disconnected. Active terminals are forcibly logged off. Other (that is, non-VSE/ICCF) transactions can be started in the CICS/ICCF partition in the normal fashion.

A certain time prior to disconnecting the VSE/ICCF library file, the operator should issue the /WARN command, for example

```
/WARN 10
```

After the /DISCONN DTSFILE command has been processed, the following message appears:

```
K131I DTSFILE IS DISCONNECTED
```

The operator can now start the DTSUTIL or DTSANALS utilities in a batch partition to do backup, restore, or other maintenance on the VSE/ICCF library file.

USER userid
Causes a particular terminal user to be logged off.

TERM termid

Causes the terminal user at terminal 'termid' to be logged off. The TERM operand is useful when, for example, the terminal user has just logged on but the logon is still in process so that the terminal user is not yet known to VSE/ICCF. To know which users/terminals are currently active, issue the /USER command.

Note:

1. The /DISCONN command with the user or terminal option should mainly be used in case of terminal or line problems. No warning message is therefore displayed on the affected terminal.

   An example of the use of the command is a logon reject, which may occur, for instance, when a session previously logged on at a terminal has not come to an orderly end. VSE/ICCF informs you in this case by the (unnumbered) message

   ```
   *TERMINAL NOT FREE – DISCONNECT FOR USER userid IS REQUIRED
   ```

   VSE/ICCF cannot process a log-on at this terminal until your operator has issued the command

   ```
     /DISC USER userid
   ```

   The terminal user may log on again, immediately after the forced logoff is completed; either at the same terminal as before (after the problem is resolved) or at another terminal. The successful completion of this type of command is not indicated; to see whether the user has been logged off successfully, the operator has to submit the /USER command.

2. If the /DISCONN command cannot complete because of active or suspended VSE/ICCF transactions (ICCF, I$$1-I$$8, I$$P), terminate such a transaction with the appropriate CEMT or CSMT transaction of CICS Transaction Server for example

   ```
     CSMT TRMNAT,YES,tasknumber
   ```

   After successful termination of the transaction, the /DISCONN command will complete.

3. The /DISCONN command with the user or terminal option can be used while the DTSFILE is being disconnected.

4. After a CICS Transaction Server autoinstall terminal that has a VSE/ICCF session going is set out-of-service, a subsequent /DISC USER or /DISC TERM will not work. This is because the terminal related TCT entry is no longer available to CICS Transaction Server. You must reattach the terminal to CICS Transaction Server and then issue the /DISC USER or /DISC TERM command to force a logoff of the user.

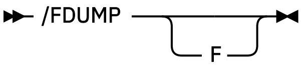
/DISPLAY Command

►►─ /DISPlay ─►◄

This command displays the status of the background (interactive partition) portion of VSE/ICCF. The following information is displayed:

- Number of active request queue entries and total request queue entries.
- Number of active interactive partitions and total number of interactive partitions.
- Number of active background tasks and total number of background tasks.
- One line for each active interactive partition indicating the partition number, the current user, the job status code, the partition identification code, and a short description of the job status.

/FDUMP Command



This command prints VSE/ICCF control tables on SYSLST in the same manner as the utility DTSFDUMP does.

F
With this operand specified, the CICS/ICCF partition below 16M will be dumped in addition.

The command takes a while till completion. Watch for the appearance of message

K132I DUMP COMMAND COMPLETED

The output is sent to SYSLST. The VSE/POWER JECL statement LST should have RBS=nnn specified to force segmentation of the output. Depending on the operand and the size of the CICS/ICCF partition the list may contain between 100 and more than 1000 pages.

Note that if another CICS function is using SYSLST at the time the dump takes place, the output might be intermixed.

For further details on the /FDUMP function, refer to section "Dump-Formatting (DTSFDUMP) Utility Program" on page 118.

/ICCFEND Command

```
▶▶─ /ICCFEND ─▶◀
```

This command is the standard command for terminating VSE/ICCF while keeping CICS active. All active VSE/ICCF users are disconnected, the DTSFILE is closed and the CICS GETVIS storage occupied by VSE/ICCF is freed. See also "Shutdown Procedure" on page 75.

You can restart VSE/ICCF at any time by running the transaction I$ST.

/MAP Command

```
▶▶─ /MAP ─▶◀
```

This command causes one line to be displayed for every interactive partition defined within VSE/ICCF The following information is displayed:

Partition number and identifier
Scheduling class(es)
Virtual address of the start of the interactive partition
Size of the interactive partition
The user who owns the interactive partition
Number of pre-allocated work files for the partition

/PDUMP Command

```
▶▶─ /PDUmp ─▶◀
```

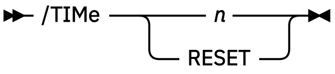
The use of the /PDUMP command will force a partial dump on SYSLST, provided SYSLST is assigned (but not to a dummy device). The following areas are printed: common systems area, request queue, partition blocks, task control blocks, file control areas, partition block extensions including interactive partition comregs.

/SEND Command

```
▶▶─ /SENd ─┬─ ALL ────┬─ message ─▶◀
           └─ userid ─┘
```

This command allows you to send messages to individual terminal users. The userid is the name under which the user is defined to VSE/ICCF. If ALL is specified, the message is sent to all users who are currently logged on. The command plus the message text may be up to 72 characters long.

/TIME Command



The /TIME command causes the high priority task timer to be set to a high value. Use the command only for debugging purposes (to run SDAID of the VSE system) or for reasons of performance measurement, that is, when you do not want the high priority task to execute frequently. Because of the high timer value, the interactive partition buffers at end-of-job do not appear on the terminal screen until the high priority task's timer elapses. Therefore, reset the timer to its generated value if you want to terminate the interactive partition in which you were debugging a program.

As to performance measurements, use the /TIME command only for measuring foreground functions.

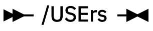
If you want to tune CICS Transaction Server while it runs with VSE/ICCF, using the monitoring facilities of CICS Transaction Server, you should:

1. Issue the /TIME command to increase the high priority task timer value.
2. Avoid that processing in interactive partitions is started.

n
Is the timer value in minutes. The smallest allowed value is 5. The largest value is 900.

RESET
Causes the high priority task timer to be reset to its generated value (the same happens if you specify an invalid operand).

/USERS Command

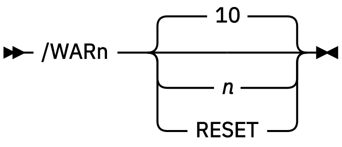


This command causes a list of all VSE/ICCF users to be displayed at the console. One line is displayed for each VSE/ICCF user. This line contains the user identification code, the terminal identification code and the user's current status. The current status is displayed as a number which is to be translated as follows:

```
     00     Command mode 'CM'
     01     Input mode 'IN'
     02     List mode 'LS'
     04     Edit mode 'ED'
     08     List mode 'LS'
     16     Execution mode 'EX'
     17     Execution mode 'EX'
     18     Spool mode 'SP'
     19     Conversational read 'RD'
```

If '****' is displayed in the user field, logon is still in progress and the user is not yet known to the system.

/WARN Command



Use the command approximately fifteen to thirty minutes before you intend to terminate VSE/ICCF. It causes the

```
*PLEASE LOG OFF*
```

message to be displayed with each terminal response at each VSE/ICCF user's terminal.

n
Is a period of time in minutes. After this time, no new interactive partition executions may be started. The smallest allowed value is 10, the largest value is 30. A default time of 10 minutes is assumed if the operand is omitted or not specified according to the syntax.

RESET
Causes the intended shutdown to be canceled and to switch back to normal processing.

For more information on how to perform a correct system shutdown, refer to "Shutdown Procedure" on page 75.

Leaving the Console Clear

Instruct your operator to respond as soon as possible to a VSE/ICCF message that needs a reply. The entire VSE/ICCF environment might be suspended if, for example, a task of VSE/ICCF in the system's logical transient area issues such a message. In this case, all programs that need the logical transient area have to wait until the operator has replied to the message.

Abnormal End of VSE/ICCF Operation

Occasionally it may be necessary to abnormally end VSE/ICCF because a problem occurred.

Before you end VSE/ICCF operation, take a dump of the CICS/ICCF partition: issue the VSE operator command DUMP, specifying the partition in which VSE/ICCF is active.

If the VSE/ICCF operator communication routine can still be accessed, terminate VSE/ICCF by entering the command /ICCFEND or by calling transaction I$SH. If you cannot gain access to the operator communication routine, you must cancel the partition by issuing the VSE command CANCEL via the attention routine. Consult the section "Backup and Recovery Procedures" on page 76 if it should be necessary to abnormally terminate the system.

Shutdown Procedure

Just as a proper procedure should be followed when starting VSE/ICCF, so a proper procedure should also be followed for terminating the system. Following the correct shutdown procedure will guard against improper termination of important functions and ensure that temporary disk areas are freed. More important, it will give terminal users time to finish updates or save input areas.

The following shutdown procedure is recommended:

1. Fifteen to thirty minutes prior to shutdown, issue the /USERS and/or /DISPLAY commands (see section "VSE/ICCF Operator Commands" on page 70). If no terminal users are logged on, it may be possible to terminate the system without the warning steps below.
2. Assuming users are still working with the system, issue the /WARN command fifteen to thirty minutes prior to shutdown. The terminal users will receive the *PLEASE LOG OFF* message with each terminal response after the /WARN is issued. In the /WARN command, specify a time value that allows background activities to be completed by the time VSE/ICCF terminates. For example, if you intend to shut down in 20 minutes, type /WARN 15 to prevent new executions from being started during the last 5 minutes.
3. At the specified shutdown time, reissue the /USERS command to verify that all users have logged off. If some users have still not logged off, it is a matter of installation practice whether they are to be notified verbally or whether they are just logged off.

4. Finally VSE/ICCF must actually be terminated. Enter the command /ICCFEND or start the transaction I$SH which will immediately terminate VSE/ICCF.

Procedures for Connecting/Disconnecting DTSFILE

You may quiesce VSE/ICCF without shutting down CICS Transaction Server. You simply disconnect the VSE/ICCF library file by issuing the /DISCONN DTSFILE command. A few minutes prior to this command, give a /WARN to the terminal users because the /DISCONN command will forcibly log them off. After the library has been disconnected, you may, from another partition, do backup, restore, or other maintenance work on the VSE/ICCF library file. At the same time, other (that is, non-VSE/ICCF) transactions can be started in the normal fashion.

You reconnect the disconnected VSE/ICCF library file by issuing the /CONNECT DTSFILE command.

Under the following circumstances VSE/ICCF automatically leaves the VSE/ICCF library file in a disconnected state during startup:

1. VSE/ICCF recognizes that the last shutdown had been abnormal (caused, for example, by a power failure), in which case the operator receives the following messages

   ```
   K008I   ICCF LIBRARY MAY HAVE BEEN DESTROYED - RUN DTSANALS
           IN ANOTHER PARTITION
   K131I   DTSFILE IS DISCONNECTED
   ```

   To recover the DTSFILE, the operator must run DTSANALS in another partition.
2. VSE/ICCF recognizes that the DTSFILE is locked by a write access from a VSE/ICCF utility program running in another partition. The operator receives the following messages:

   ```
   K099I   ICCF COULD NOT GET EXCLUSIVE CONTROL OVER DTSFILE, RTNCODE=xx
   K131I   DTSFILE IS DISCONNECTED
   ```

VSE/ICCF is started, but terminal users are not allowed to log on to VSE/ICCF. CICS Transaction Server is already up at this time, and non-VSE/ICCF transactions may be started from the terminals. When the cause of the message has been removed, the operator completes the startup of VSE/ICCF by issuing the command /CONNECT DTSFILE.

Backup and Recovery Procedures

To ensure that the VSE/ICCF environment runs smoothly and to minimize loss of users' data, VSE/ICCF offers backup and recovery facilities. There are basically three situations when you need these facilities:

1. To recover from I/O malfunctions.
2. To reorganize the VSE/ICCF library file to improve performance.
3. To recover after a system failure.

The first two situations are covered by the BACKUP and RESTORE functions of DTSUTIL, the third one by the RECOVER function of DTSANALS.

Backing Up and Restoring the VSE/ICCF Library File

If the contents of your VSE/ICCF library file happens to be destroyed (due to an I/O malfunction, for example), you must restore the file from your most recent backup.

All changes to the file between the last backup and the current malfunction are lost. It is therefore most beneficial to create a backup of the VSE/ICCF library file as often as possible.

During backup, the accounting information relating to user disk storage usage will be validated and updated if necessary. For this reason the backup job should always be run before punching and clearing of the accounting data.

Note: You can run a backup or restore job only when the CICS/ICCF partition is down or, if that partition is up, when the VSE/ICCF library file is disconnected.

Figure 13 on page 77 shows a job stream which initiates a backup. The first job writes the library-file backup onto tape, the second job onto disk.

Figure 14 on page 77 shows two sample jobs that restore the VSE/ICCF library file. The first job restores the file from a backup on tape, the second one from a backup on disk. (DLBL/EXTENT information for the DTSFILE is not shown because it is assumed to be stored in the standard label information area.)

Instead of creating your own jobs, you can use z/VSE dialogs to create backup and restore jobs. How to work with these dialogs is described in z/VSE Operation.

```
// JOB    BACKUP VSE/ICCF FILE TO TAPE
// TLBL   DTSBKUP,'ICCF BACKUP'
// ASSGN  SYS005,cuu  (tape drive)
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC   DTSUTIL
BACKUP
/*
/&
// JOB    BACKUP VSE/ICCF FILE TO DISK
// DLBL   DTSBKUP,'ICCF BACKUP',99/365
// EXTENT SYSnnn,vol-id,type,seq,begin,end
// ASSGN  SYSnnn,cuu   (disk drive)
// EXEC PROC=DTRICCF            (assign DTSFILE)
// EXEC    DTSUTIL
BACKUP
/*
/&
```

Figure 13. Sample Jobs — Backup of the VSE/ICCF Library File

```
// JOB    RESTORE        FROM TAPE
*  THIS ASSUMES THAT FORMATTING IS NOT REQUIRED
// TLBL   DTSRSTR,'ICCF BACKUP'
// ASSGN  SYS004,cuu          (assign backup tape)
// EXEC PROC=DTRICCF          (assign DTSFILE)
// EXEC   DTSUTIL
RESTORE          (Reply GO when console message appears)
/*
/&
// JOB    RESTORE       FROM DISK
// DLBL   DTSRSTR,'ICCF BACKUP'
// EXTENT SYSnnn,vol-id,type,seq,begin,end
// ASSGN  SYSnnn,cuu          (assign backup disk)
// EXEC PROC=DTRICCF          (assign DTSFILE)
// EXEC   DTSUTIL
RESTORE          (Reply GO when console message appears)
/*
/&
```

Figure 14. Sample Jobs — Restore the VSE/ICCF Library File

Reorganizing the VSE/ICCF Library File

Reorganize your VSE/ICCF library file before the number of high file records reaches zero. This ensures that users get the best possible response time.

A reorganization places all related records in contiguous areas, thus reducing file accesses. Also, the free chain records are removed from the file which results in two additional benefits. First, the file data is more compact and disk arm movement is reduced. Secondly, new records are allocated from the high file area rather than from the free chain, which reduces file accessing. In addition, directory records are placed adjacent to one another so that directory scans are faster.

Message K088I during the startup of VSE/ICCF indicates how close the number of high file records is to zero. A zero percent means that the high file area is used up and new records are allocated from the free chain which results in poorer performance.

To reorganize the VSE/ICCF library file, combine one of the above backup job streams with the appropriate restore job stream. Again, keep in mind that a backup or restore cannot be performed while VSE/ICCF is active.

Recovery After System Failure

If one of the following occurs after VSE/ICCF has been initiated and used, the RECOVER function of DTSANALS must be run:

- Processor power failure.
- Processor power off without VSE/ICCF being terminated.
- Machine check or program check in supervisor requiring re-IPL.
- Termination of the VSE virtual machine under VM without VSE/ICCF termination.
- Any other system termination such that the VSE/ICCF normal EOJ routine or ABEND STXIT routine (message K017I) is not completed.

When one of the above occurs, it is likely that there are unwritten file buffers in VSE/ICCF storage. Thus, it is also likely that chains are broken or misdirected on the library file. If this condition is not corrected using the RECOVER function of DTSANALS, it is likely that the file errors will be compounded the next time VSE/ICCF is used. This can result in severe problems, loss of data or an inability to back up the file.

Thus, you are forced to run this function whenever the system terminates in some way other than normal shutdown or via a soft ABEND such that the message K017I ICCF ABEND PROCESSING COMPLETE appears. Next time VSE/ICCF is started the DTSFILE will be disconnected automatically. Thus, it is not possible to work with VSE/ICCF without prior recovery of the DTSFILE.

It is also beneficial (but not imperative) to run the RECOVER function after a soft ABEND (where message K017I appears). This will recover certain file work areas and reconnect them to the free chain. You can run a recover job only when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected.

Figure 15 on page 78 shows a sample job.

```
// JOB RECOVER AFTER FAILURE
// ASSGN SYS001,cuu        |  (the standard compiler work
// ASSGN SYS002,cuu        |  files are sufficient)
// DLBL IJSYS01,,0
// EXTENT SYS001,volser,1,0,addrs,trks
// DLBL IJSYS02,,0
// EXTENT SYS002,volser,1,0,addrs,trks
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSANALS
RECOVER
/*
/&
```

Figure 15. Sample Job — Recover Run after an Abnormal Termination

Chaining Errors

If the RECOVER function of DTSANALS finds chaining errors, check the listing to determine whether the errors are important. That is: check whether good user data has been deleted and attached to the free chain.

Most chaining errors affect temporary areas and are therefore not critical; for example a user's input area was not saved or freed. These errors can generally be recognized by the nature of the data concerned. That is, object decks or print areas will have a 1 or 2 in column 1 of each record. Uninitialized print and punch areas will have the characters /* */ in columns 1 to 5.

If it appears that valid user data has been erroneously freed, several tools within the DTSANALS utility can be used to isolate, display and reattach the data to a library member. First, find the first and last relative record number of the data to be reinserted into the library member. These numbers, which are represented in hexadecimal, may be determined by displaying the pertinent section of the free chain, using DTSANALS. For example:

```
// JOB  PRINT PORTION OF FREE CHAIN
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSANALS
CHASE   FREE
LOCATE  /unique string in first record to save/
```

```
PRINT   300                NO. OF RECORDS TO BE PRINTED
/*
/&
```

Assume that, from the printout, the first record number of the data to be reinserted is 74AC and the last is 63F3. Assume also that the data is to be reinserted at the bottom of the library member named LIBDATA in library number 3:

```
// JOB  RECOVER DATA FROM FREE CHAIN
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSANALS
LIB     3
MEMBER   LIBDATA
BOTTOM
INSERT   74AC,63F3
/*
/&
```

Note: You can run this job only when VSE/ICCF is down or when the VSE/ICCF library file is disconnected.

Correcting the Free Chain

If, after a RECOVER run of DTSANALS, it appears that the free area chain (all imbedded free records up to the last used record in the file) is shorter than it should be, it is possible that a chaining error has tied the free chain into a library member. To correct this problem first establish the start of the free chain, which means finding the member to which the free chain is incorrectly attached.

Scan the listing from the recovery run for a member count which is unreasonably large. All members whose record counts exceed 3000 will be printed on the listing. If none exists, the free chain is probably correct and the file is simply full. If you find an oversize member, however, print it to see if it contains records which logically do not belong to it.

Assume that the suspected member is in library number 5 and is named JACDATA:

```
// JOB  PRINT A MEMBER
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSANALS
LIB     5
MEMBER  JACDATA
PRINT   ALL
/*
/&
```

If the printed member is at fault, reattach the extraneous records to the free chain in the following way:

```
// JOB  CORRECT FREE CHAIN
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSANALS
LIB     5
MEMBER  JACDATA
LOCATE  /unique string in the last good record/
NEXT    1
DELETE  99999                (delete the remaining records)
/*
/&
```

Deleting all of the remaining records from the member will cause these records to be returned to the free chain.

Note: You can run this job only when VSE/ICCF is down or when the VSE/ICCF library file is disconnected.

Adding or Updating Mail

You can communicate with VSE/ICCF terminal users via the MAIL facility. This facility allows the VSE/ICCF administrator to convey general information to all users, or specific information to certain users as specified by their user identifications.

The /MAIL Command

The terminal user has access to MAIL information by issuing the /MAIL command, which returns a display to the terminal of all general mail and any specific mail associated with the user's identification.

The A$MAIL Common Member

All general notices or mail and all specific user mail is contained in a VSE/ICCF library member named A$MAIL. This library member should reside in the common library. Thus, it is accessible to each VSE/ICCF terminal user. When the terminal user issues the /MAIL command, it is the A$MAIL library member which is displayed.

Mail Format

All general mail should logically be first in the A$MAIL library member, after which the specific user mail should follow. All general mail consists of 80 column card images with blanks in columns one through four. All specific user oriented mail should have the user identification in columns one through four of each mail record.

Initial Mail Setup

The instructions for setting up A$MAIL are given here. (DLBL/EXTENT information for the DTSFILE is not shown because it is assumed to be stored in the standard label information area.)

```
// JOB SETUP A$MAIL
// EXEC PROC=DTRICCF           (assign DTSFILE)
// EXEC DTSUTIL
ADD MEMBER (2,A$MAIL,AAAA)
    ***VSE/INTERACTIVE COMPUTING AND CONTROL FACILITY***
       GENERAL NOTICES AND MAIL
    ... |                     Any general information for
    ... |                     all users
    ... |
    ***BEGIN SPECIFIC USER MAIL***
GNCB SORRY JACK, CAN'T MOUNT YOUR PACK ABEL00
GNCB DUE TO BAD SURFACES.
INFA THE BITON, BITOFF AND TESTBIT SUBROUTINES
INFA HAVE BEEN ADDED TO THE SYSTEM.  THESE ROUTINES
INFA MAY NOW BE USED AS DESCRIBED IN LETTER
INFA 25A-79-074.
END OF MEMBER
/*
/&
```

In this example, the VSE/ICCF administrator's identification (AAAA) is used and the mail file is stored in the common library (in this case, library 2).

Updating Mail

There are two ways to update A$MAIL. The first is by using punched cards in an offline manner. Simply add your changes to the A$MAIL deck above and then run the following job before you start VSE/ICCF:

```
// ASSGN SYS010,cuu
// DLBL DTSFILE,'ICCF.LIBRARY',99/365,DA
// EXTENT SYS010,volserno,1,0,addr,number
// EXEC DTSUTIL
PURGE LIB 2 MEMBER (A$MAIL)
ADD MEMBER (2,A$MAIL,AAAA)
  ...                |
  ...                |        Updated mail deck
  ...                |
END OF MEMBER
/*
/&
```

Note: You can run this job only when VSE/ICCF is down or when the VSE/ICCF library file is disconnected.

A second way of updating the A$MAIL library member is available to the VSE/ICCF administrator (or anyone else whose user profile permits updating of common data and who has the use of a terminal). This way involves using the editing facilities of VSE/ICCF to update the A$MAIL member online while VSE/ICCF is running.

Chapter 4. Submit-To-Batch Program

This topic describes the VSE/ICCF Submit-to-Batch (DTSSUBMT) program.

Submit to Batch (DTSSUBMT) Program

The DTSSUBMT program forms the basis for the submit-to-batch facility of VSE/ICCF. The program's main function is to read VSE/ICCF or VSE job streams from the terminal user's library and to transfer these job streams to VSE/POWER for execution in a VSE batch partition. The job may be submitted to a system other than the one from where it is submitted.

If you want to take advantage of the VSE/POWER NOTIFY support, specify NTFYMSG=n in the POWER generation macro for VSE/POWER table generation.

Submit Procedure

The DTSSUBMT program is invoked by the SUBMIT procedure. When setting up your SUBMIT procedure, you should be aware of some restrictions with respect to the submit-to-batch facility. These restrictions, and also the use of the procedure are described in the VSE/ICCF User's Guide.

The terminal user enters the SUBMIT command with either of these options: DIRECT and RETURN. Variations on these two types of submission can be made by using other names, provided that the first letter is D or R. See “Modifying the SUBMIT Procedure” on page 86.

- The DIRECT option implies that the job is to be run and all printed output to be directed to the central system printer.
- The RETURN option implies that the print output from the job is to be held in a VSE/POWER output queue. The terminal user may then view it and manipulate it using the /STATUSP, /LISTP, /ERASEP, /CTLP and /ROUTEP system commands, or the GETL procedure.

DIRECT is the default option.

DTSSUBMT Program Parameters

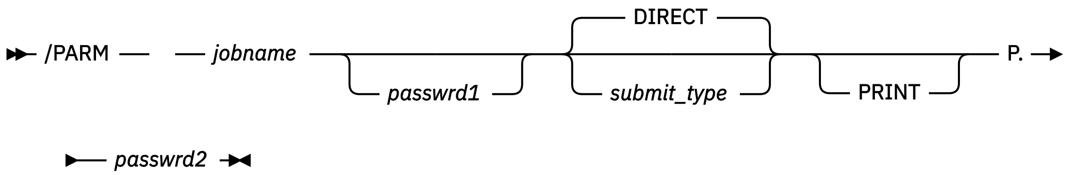
By alterations to the SUBMIT procedure, the VSE/ICCF administrator can:

- Specify JECL defaults for submitted jobs.
- Control to which partition the jobs are submitted.
- Control to which system the jobs are submitted.
- Restrict certain JECL operands so that the default cannot be overridden.
- Specify the default SUBMIT option.
- Add additional options for the type of submission.

The DTSSUBMT program reads a /PARM statement containing the options for submission. The program also reads two sets of VSE/POWER JECL statements. These two sets correspond to the two options DIRECT and RETURN and any installation-applied options.

/PARM Statement

The /PARM statement, if present, must be the first record read by DTSSUBMT. It has the following format:



jobname
: Is the name of the library member that contains the job stream. It is also the name by which it will be known to VSE/POWER if no * $$ JOB statement with a valid job name is included in the submitted member.

passwrd1
: Is the password that is associated with the member.

submit-type
: Either DIRECT or RETURN. If omitted, DIRECT is assumed.

PRINT
: Causes the JCL, as passed to VSE/POWER, to be displayed at the submitter's screen.

passwrd2
: Is the password by which this job is protected under VSE/POWER.

The /PARM statement must contain 'jobname' and 'passwrd2'. The /PARM statement may be omitted by users who are authorized to submit directly to VSE/POWER without the use of the IBM-supplied standard SUBMIT procedure (user profile OPTA bit 7 on, as described in section "Adding or Changing a User Profile" on page 128). In this case the submit type is DIRECT, and the job name is derived from the user identification and the terminal identification.

/UPSI Job Entry Statement

Certain DTSSUBMT options are controlled by UPSI statement switches as follows:

a = 0 –
: Nothing special

1 –
: Print VSE/ICCF JES contained in input stream

b = 0 –
: Nothing special

1 –
: Print VSE job control and VSE/POWER JECL

c = 0 –
: Nothing special

1 –
: Bypass output to VSE/POWER (used while setting up and 'testing' the SUBMIT procedure)

d = Reserved

e = 0 –
: Use the VSE linkage editor

1 –
: Passed to VSE unchanged

f = 0 –
: Place assignments for unit record devices in job stream

1 –
: Omit assignments for unit record devices when changing VSE/ICCF JES to VSE JCL

g = 0 –
: Nothing special

1 –
Special EOF record /* */ is recognized by DTSSUBMT

h = Reserved

VSE/POWER JECL Statements

There are two sets of VSE/POWER JECL statements which are read by the DTSSUBMT program, one for submit-type DIRECT and another for submit-type RETURN.

These statements are the standard VSE/POWER JECL statements JOB, LST, PUN, EOJ. They are supplied by IBM as part of the SUBMIT procedure. The DTSSUBMT program uses them as a model to build VSE/POWER JECL for the output job stream. The end of these model statements is indicated by a non-VSE/POWER statement with XXXXXXXXXXXXXXX beginning in column 1. It must be used to separate the model JECL from the data which the user wants to submit.

If one of the two sets of model statements is omitted, the option it corresponds to (DIRECT or RETURN) is not allowed. Any job requesting that option will be canceled.

The model statements have the standard VSE/POWER format. The values in the keyword operands of these statements are one of these types:

Excluded from user specification
Overriding the user specification
Defaults

You recognize the type of a value by its format in the model statement.

An operand that has a value of '***' is excluded from user specification; for example, XDEST=*** in one of the * $$ JOB statements. Only authorized users may specify those operands. Authorization is established through bit 7 of OPTA in the user profile (for a description of the OPTA operand, see section "Adding or Changing a User Profile" on page 128).

An override value takes precedence over a value for the same keyword submitted by the user.

Default values are indicated by enclosing the value in parentheses (multi-option values have double parentheses). A default value is supplied for the keyword only if that keyword is omitted by the user.

The following examples illustrate the differences:

```
* $$ LST  CLASS=Q,DISP=K,PRI=(8),RBM=((1000,100))
```

In this example, CLASS=Q and DISP=K will be forced regardless of whether the user specifies these operands in the job stream. On the other hand, PRI=8 and RBM=(1000,100) will only be included on the submitted LST statement if the user has not included his own PRI and RBM operands. Thus, if the user's LST statement was as follows:

```
* $$ LST  CLASS=X,DISP=H,PRI=6,COPY=5
```

then the following would be submitted to VSE/POWER:

```
* $$ LST  CLASS=Q,DISP=K,PRI=6,COPY=5,RBM=(1000,100)
```

Here is another example:

```
Model:        * $$ PUN  DISP=(D),CLASS=(A)
User:         * $$ PUN  DISP=H
To VSE/POWER: * $$ PUN  DISP=H,CLASS=A
```

Any valid keyword operands which the user specifies in the input and which are not specified in the model statements will be passed to VSE/POWER. Invalid keyword operands and user defined keywords are ignored.

Note:

1. If bit 7 of the OPTA option byte is set on for a user (see section "Adding or Changing a User Profile" on page 128), this user's specifications in the VSE/POWER JECL are not overridden by the SUBMIT procedure values. If bit 7 of the OPTA byte is set off for a user and the user did specify a jobname in the VSE/POWER JECL, this jobname will be replaced by the VSE/ICCF library member name (as long as the default specification for JNM in the IBM-supplied SUBMIT procedure is used).
2. The DTSSUBMT program does not transfer columns 73 through 80 of a VSE/POWER JECL statement (unless the user's OPTB bit 7 is set on and columns 73 through 76 of the statement contain the characters $SLI, in which case DTSSUBMT transfers all 80 columns unchanged).
3. If bit 1 of OPTC is off (which is the normal case), VSE/ICCF always inserts an * $$ LST Statement and an * $$ PUN statement at the beginning of the submitted job stream (for a description of the OPTC operand, see section "Adding or Changing a User Profile" on page 128). If bit 1 of OPTC is on and the XDEST operand is specified in the submitted * $$ JOB statement, VSE/POWER JECL statements are not inserted. This provides the ability to submit job streams unchanged to non-VSE/POWER nodes.

The operand 'jobname' has special significance on the selection of the model statement. The specified name indicates the type of submission that the JECL is modeling. Default and override values may be different for each set of the model statements. Job names can have the following meaning:

DIRECT
Submission is directly to VSE/POWER. No print data from the execution is available to the user. The output disposition is usually D or H and the class is decided locally, but is normally 0 for BG and A for any other partition.

RETURN
Submission is directly to VSE/POWER. The print data is to be left in the VSE/POWER print queue for later review at the terminal. The print disposition should be specified as K. The class of the printed output should be equal to the value in the CRJE operand of the DTSOPTNS macro (the default is Q).

Modifying the SUBMIT Procedure

Figure 16 on page 87 shows the SUBMIT procedure that is supplied with VSE/ICCF. This procedure can be modified to conform to the standards at your location. Types of possible modification are discussed after the following figure.

```
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
*  PROCEDURE TO SUBMIT A MEMBER DIRECTLY TO POWER
* SUBMIT NNNN (PASS1) (DIRECT/RETURN) (PRINT) (PWD=PASS2)
*        (ICCFSLI)
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
&&OPTIONS 0010011
&&IF &&PARMCT NE 0 &&GOTO START
&&TYPE ENTER   NAME (PASS1) (DIRECT/RETURN) (PRINT) (PWD=PASS2)
&&TYPE         (ICCFSLI)
&&READ &&PARAMS
&&IF &&PARMCT EQ 0 &&EXIT
&&LABEL START
&&SET &&COUNT1 &&PARMCT               SET LOOP COUNT
&&SET &&VARBL6 &&PARAM1               SET MEMBER NAME
&&SHIFT 1                             SHIFT OUT MEMBER NAME
&&LABEL LOOP
&&SET &&COUNT1 *-1                    SUBTRACT LOOP-COUNT
&&IF &&COUNT1 EQ 0 &&GOTO LIST        ALL PARAMETERS SCANNED?
&&IF &&PARAM&&COUNT1 NE ICCFSLI &&GOTO 3
&&SET &&VARBL5 'ICCFSLI''             SET 'ICCFSLI'
&&GOTO SHIFT                          SHIFT OUT ICCFSLI
&&IF &&PARAM&&COUNT1 EQ DIRECT &&GOTO SHIFT
&&IF &&PARAM&&COUNT1 NE PRINT &&GOTO 3
&&SET &&VARBL1 01                     SET 'PRINT'
&&GOTO SHIFT
&&IF &&PARAM&&COUNT1 NE RETURN &&GOTO 3
&&SET &&VARBL7 'RETURN'               SET 'RETURN'
&&GOTO SHIFT
&&IF &&PARAM&&COUNT1 EQ SEGMEMT &&GOTO SHIFT   NOT SUPPORTED
&&SET &&VARBL4 &&SUBSTR 1 4 &&PARAM&&COUNT1
&&IF &&VARBL4 NE PWD= &&GOTO 3
&&SET &&VARBL3 &&SUBSTR 5 8 &&PARAM&&COUNT1  SET 'PWD=PASSWORD'
&&GOTO SHIFT
&&SET &&VARBL0 &&PARAM&&COUNT1        SET PASSWORD
&&LABEL SHIFT
&&SHIFT &&COUNT1                      SHIFT OUT PARAMETER
&&GOTO -LOOP
&&LABEL LIST
/LIST 1 1 &&VARBL6 &&VARBL0;
&&IF &&RETCOD NE *FILE &&GOTO 3
&&TYPE *MEMBER &&VARBL6 NOT IN LIBRARY OR EMPTY
&&EXIT
&&IF &&RETCOD NE *INVALID &&GOTO 3
&&TYPE *INVALID PASSWORD OR INVALID MEMBER NAME &&VARBL6
&&EXIT
```

```
                                                     Column 72 ───┐
                                                                  |
                                                                  V
&&IF &&RETCOD NE *MISSING &&GOTO 5
&&TYPE *ENTER PASSWORD FOR MEMBER &&VARBL6
&&READ &&VARBL0
&&IF &&VARBL0 NE ' ' &&GOTO -LIST
&&EXIT
&&SET &&VARBL9 '* $$'
&&SET &&VARBLA '* $$ JOB'
/INP NOPROMPT
&/LOAD DTSSUBMT
&&IF &&VARBL1 EQ 01 &&GOTO 3
&/OPT JSDATA CONTINUE
&&GOTO 2
&/OPT JSDATA
&/UPSI &&VARBL1                                                       (1)
/PARM &&VARBL6 &&VARBL7 DIRECT P.&&VARBL3                             (2)
&&VARBLA JNM=DIRECT,DISP=D,CLASS=A,NTFY=YES,                      X   (3)
&&VARBL9 XDEST=***,LDEST=***,PDEST=***
&&VARBL9 LST DISP=D,CLASS=A,DEST=*
&&VARBL9 PUN DISP=(D),RBM=(1000),CLASS=(A),DEST=*
&&VARBL9 EOJ
&&VARBLA JNM=RETURN,DISP=D,CLASS=A,NTFY=YES,                      X   (4)
&&VARBL9 XDEST=***,LDEST=***,PDEST=***
&&VARBL9 LST DISP=K,CLASS=Q,RBS=(0),DEST=***
&&VARBL9 PUN DISP=K,CLASS=Q,RBS=(0),DEST=***
&&VARBL9 EOJ
XXXXXXXXXXXXXXX
&/INCLUDE &&VARBL6 &&VARBL0 &&VARBL5
&&LABEL NONAME
&&IF &&RETCOD NE *LIBRARY &&GOTO NOFULL
&&TYPE *LIBRARY OR LIBRARY DIRECTORY FULL
&&EXIT
&&LABEL NOFULL
/END
/PEND
/RUN
END OF MEMBER
```

Figure 16. IBM-Supplied SUBMIT Procedure

- Controlling printing of job control (see reference (1) in Figure 16 on page 87). The /UPSI job entry statement is used to indicate the installation preference for printing of input and output control statements. See section "/UPSI Job Entry Statement" on page 84 for the meaning of the UPSI switches.
- Changing the default submission type (see reference (2) in Figure 16 on page 87).

The third operand of the /PARM statement is the default for the submission-type option. That is, if a user of the system does not specify an option, this default is used. The &&VARBL3 after the P. represents the password through which the job is protected under VSE/POWER.

- Controlling the operands of VSE/POWER JECL statements (see the references (3) and (4) in Figure 16 on page 87). Information is supplied by sets of model JECL following the /PARM statement. The models show:
  - Which user-specified values are overridden.
  - The values that serve as default.
  - The values that the user can specify.

  You can alter these two sets of JECL to control the defaults and overrides for the two submission options DIRECT and RETURN.

  As the compare is on the full VSE/POWER name, multiple sets of JECL can be used as long as the name begins with either a D or R. For example, JNM=DIRECT1, JNM=DIRECT2 and JNM=DIRECT3 are all valid and each could have a different set of JECL.

  If you want to alter the DIRECT or RETURN model statements, make sure that:
  - The value assigned to the DISP operand in the * $$ JOB statement and the value assigned to the DISP operand in the * $$ LST and * $$ PUN statement of the DIRECT model statements reflect the disposition of the CRJE operand ('z') of DTSIGEN. If no user statement * $$ LST DISP= is provided and the model statement LST DISP= has been removed, LST DISP will still default to K.
  - The value assigned to the CLASS operand in the * $$ JOB statement reflects the class of the CRJE operand ('r') of DTSIGEN;
  - The value assigned to the CLASS operand in the * $$ LST statement of the DIRECT model statements reflects the class of the CRJE operand ('y') of DTSIGEN;
  - The value assigned to the CLASS operand in the * $$ LST statement of the RETURN model statements reflects the class of the CRJE operand ('x') of DTSIGEN.

  For more information on how these model statements might be altered to adhere to installation standards, refer to “DTSSUBMT Program Parameters” on page 83 and “VSE/POWER JECL Statements” on page 85.

The ICCFSLI Operand of the /INCLUDE Statement

The job stream which a terminal user submits to VSE/POWER may contain one or more /INCLUDE statements (see “The INCLUDE Facility” on page 16). By specifying the ICCFSLI operand in the /INCLUDE statement, the terminal user directs VSE/POWER to read the member from the VSE/ICCF library file. Refer to the " /INCLUDE" job entry statement in the VSE/ICCF User's Guide.

The DTSSUBMT program transforms the /INCLUDE statement into the VSE/POWER JECL statement

```
 * $$ SLI ICCF=(member[,password]),...
```

Without the ICCFSLI operand, the DTSSUBMT program would read the member from the VSE/ICCF library file and place it in the VSE/POWER reader queue. From there, VSE/POWER would read it into the program which is about to be executed. Thus, specification of the ICCFSLI operand reduces the input/output traffic in your system. On the other hand, use of the ICCFSLI operands is of advantage only for larger members (20 records or more). Also, the member should be left intact as long as the job has not started. In other words, if you wanted to further update the member or even delete it immediately after submission, you should submit the job without the ICCFSLI option.

The /INCLUDE statement in the IBM-supplied SUBMIT procedure does not contain the ICCFSLI operand. You may add it. If you do so, the SUBMIT program generates the * $$ SLI statement and places the necessary JECL statements around it. The job stream is not further examined by VSE/ICCF. Therefore, add the ICCFSLI operand to the SUBMIT procedure only if you are certain that your terminal users submit job streams entirely without VSE/ICCF job entry statements. If the job streams contain VSE/ICCF job entry statements, these statements must be examined by VSE/ICCF first before being transformed into VSE JCL.

When the * $$ SLI statement is being processed and the member to be included itself contains a /INCLUDE, a pertinent * $$ SLI will be generated. The depth of nesting level is unlimited. /INCLUDE statements in compressed members as well as in non-compressed members are resolved.

Before VSE/POWER includes the member, an access-control check is made if:

- The job stream is submitted from a terminal.

  The submitter must be authorized (according to the access-control rules of VSE/ICCF) to access the member and the library containing the member.
- The job stream is submitted via a 'card reader' (for example from a VM virtual reader) and the VSE sytem was IPLed with security active.

  The system requires either specification of the SEC parameter in the * $$ JECL statement or a // ID statement in the job stream. This specification identifies the submittor to the system. As a result, the normal access-control rules of VSE/ICCF apply.

  If a valid VSE/ICCF user ID is not provided or if the system was started with SEC=NO, only public members from the common library or from a public library can be included.

If the access-control check fails or the inclusion request is invalid, the job is canceled. The job is also canceled if, while a member is included, a recovery or a restore of the VSE/ICCF library file is being initiated. Inclusion of a member may be delayed if, at the time of the request, the VSE/ICCF library file is not accessible. Such a situation may occur, for example, while the library file is being recovered.

Chapter 5. Utility Programs

This topic describes the utility programs that you can use to maintain VSE/ICCF.

General Information

The utilities listed below are available to the VSE/ICCF administrator and to central site personnel whose job it is to control and maintain VSE/ICCF. The utilities carry out the following functions:

DTSANALS
Is used for library file analysis, recovery control and detailed record manipulation.

DTSAUDIT
Obtains displays of added, deleted or changed records in library members and scans a group of members, or libraries, for changes.

DTSBATCH
Executes VSE/ICCF system and context editor commands in offline mode from a 'card reader' or system console.

DTSFDUMP
Interprets and formats VSE/ICCF tables and control blocks.

DTSRELST
Generates print images from a print buffer area.

DTSUTIL
Is used to back up and restore the VSE/ICCF library file, to add or change user profiles and libraries, to add common data, and to add or purge members.

UPSI Switches for Utilities

All utilities which run in a VSE partition (as opposed to a VSE/ICCF interactive partition) use the first UPSI bit to control VSE/ICCF library activity in the following way: when this switch is set on, a write-verify will be performed after each physical write to the VSE/ICCF library file.

Files for Utilities

Following is a table that indicates which utility uses which files. The table shows what assignments are required for which utilities.

File	Description	Use	Symbolic Unit	Device Type
DTSFILE	VSE/ICCF library file	All Utilities	As defined in startup of CICS/ICCF partition	Disk
DTSBTCH	Input file	Input data file for processing by DTSBATCH	SYS004/SYSnnn	Tape/Disk
IJSYS01	Work File	Work file for analyze, recover, and reorganization processing by DTSANALS	As defined in startup JCL of CICS/ICCF partition	Disk
IJSYS02	Work File	Same as IJSYS01	Same as IJSYS01	Disk
unnamed	Input File	Tape containing a dump, which is input to DTSFDUMP	SYS007	Tape

File	Description	Use	Symbolic Unit	Device Type
DTSBKUP	Output File	Backup file for DTSUTIL backup and merge processing	SYS005/nnn	Tape/Disk
DTSRSTR	Input File	Restore file for DTSUTIL restore and input processing	SYS004/nnn	Tape/Disk
DTSMERG	Input File	Input file for merge processing by DTSUTIL	SYS004/nnn	Tape/Disk

Note: If DTSBKUP, DTSRSTR and DTSMERG are on disk, the fileid must be specified to avoid confusion of files and possible loss of data.

Size of Utilities

The minimum space required by the VSE/ICCF utilities is module size plus GETVIS space. Certain factors, however, such as the function that was requested or particular DTSFILE characteristics, may increase the space requirements.

Run Information for Utilities

The following table summarizes basic run information, in particular about the partitions where a utility can run. Details are provided in the individual utility descriptions in the following sections.

Table 3. Utilities - Run Information

Utility Name	Static/Dynamic Partition	Interactive Partition
DTSBATCH	yes, but see Note 1	no
DTSANALS	yes, but see Note 1	some commands yes, but see Note 2
DTSAUDIT	yes, but see Note 1	some commands yes, but see Note 2
DTFSDUMP	yes, run here if UPSI bit setting required see also /FDUMP	yes for online debugging of VSE/ICCF
DTSRELST	yes	no
DTSUTIL	yes, but see Note 1	some commands yes, but see Note 3 Space requirement 256K!

Note:

1. The total set of functions is supported if write access to the DTSFILE is available to the utility.
2. The set of functions is limited. The DTSFILE can be accessed for read-only.
3. The set of functions is limited. The DTSFILE can be accessed for read-only. However, the ADD/ALTER USER and ADD/ALTER LIBRARY commands can be processed.

Library File Analysis (DTSANALS) Utility Program

DTSANALS is the VSE/ICCF library file recovery and reorganization program. It is used to correct chaining errors caused by system failures, to sequentially organize free chain and directory records, to print directory listings of the libraries, and to print or punch any portion of the VSE/ICCF library file. For the use of DTSANALS following a system failure see the section "Recovery After System Failure" on page 78.

DTSANALS normally is run while VSE/ICCF is not active. If the DTSFILE has been disconnected, DTSANALS can be run even when the VSE/ICCF is active. If, however, VSE/ICCF is active *and* the DTSFILE is connected, you may only use the display, conversion, and file positioning commands of DTSANALS. In this case, commands which update record pointers (ALTER, ANALYZE, RECOVER, REORG, INSERT, DELETE, RESET) cannot be used. The display and positioning functions may even be used in an interactive partition.

Command Entry (Console versus Reader)

Commands may be given through the operator console or a 'card reader' or, if running in an interactive partition, via the user terminal. The first command will be read from the same device from which VSE job control read the EXEC command. If you want to switch to another input device later (from console to 'card reader' or vice versa), have a CONSOLE statement in your input stream or enter CARD at the console, respectively. The first time the operator console is used after card input you will be prompted with the message ENTER DTSANALS COMMAND. Requests for further commands will not include this prompting message.

Each command must begin in column 1 and must be complete on one line. Operands may continue through column 80. Multiple commands on one line are not supported. If an invalid operand is entered, the system will indicate which operand is in error and the whole command must be reentered. If input is from cards, the incorrect command will be displayed prior to the error message and you may then reply with the corrected command from the console. After the error has been corrected, the next input is read from SYSIPT.

Command Interruption

Most functions of the DTSANALS program may be interrupted via the VSE operator communication routines (by entering the VSE command MSG xx). However, none of the following functions is interruptible until it is completed or a certain point of processing has been reached. The functions are: ANALYZE, RECOVER, REORG, DELETE, DSERV, and RESET.

When, as a result of an interrupt, the console read routine received control, VSE/ICCF attempts to return to the file position established before processing for the interrupted command began. The commands which establish file position in case of an interruption are: INSERT, PRINT, LIST, COUNT and PUNCH. All other commands stop where they are interrupted, and the stop location becomes the new file position.

Work Files

Two work files are required for the DTSANALS functions ANALYZE, RECOVER and REORG. Work file names are IJSYS01 and IJSYS02. The labels and corresponding assignments for the work files must be available for DTSANALS to make the device type determination, and both work files must reside on devices of the same type. If VSAM-managed space is used for one work file, the other work file must also reside in VSAM-managed space. The space requirements vary with the VSE/ICCF library file size and the DASD type. One track or 16 FBA blocks (512 bytes per block) on each work file are sufficient for each 32K of records in the VSE/ICCF library file. If your work files are in VSAM-managed space, make sure that they will have a block size of 4100 bytes. If standard compiler work areas and assignments are available in the partition where DTSANALS is executing, there are no work file considerations.

The commands fall into seven categories: positioning, printing, mode of processing, updating, conversion, analysis and reorganization. Each of these command categories is discussed in the following sections. Individual commands are discussed following the general discussion of their category.

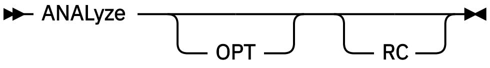
Analysis Commands

The analysis of the library file is the major function of DTSANALS. This function ensures the integrity of the members in the library and of the free areas in the VSE/ICCF library file.

The program reads all known chains of records and saves their location in a bit map. Next, it checks for any records not chained. Any that are found to be unchained are placed at the beginning of the free chain and printed so that the user can move them to a member chain, if desired.

The second phase of analysis (RECOVER) must be preceded by the first phase (ANALYZE). Any updating commands used between the two phases, or not performing the first phase specifically, will cause the first phase to be automatically initiated before the recovery phase. That is, the RECOVER command will do the ANALYZE if it needs to be done.

ANALYZE Command



OPT

Allows to make execution of the ANALYZE command conditional. If OPT is specified, the command is executed only if DTSANALS recognizes that the VSE/ICCF library file was not closed properly during the last execution of VSE/ICCF or of a VSE/ICCF utility. However, to find out whether the VSE/ICCF library file has *really* lost its integrity, you must request an unconditional (that is: without the OPT operand) run of the ANALYZE command of DTSANALS.

RC

Requests that DTSANALS provides a return code which informs about the condition of the DTSFILE.

0

DTSFILE is intact.

1

DTSFILE is intact. Some free space may not be accessible, however. At a convenient time, run backup/restore of the DTSUTIL utility to reorganize the DTSFILE.

2

DTSFILE was restored **incompletely**. The RESTORE function of DTSUTIL must be retried.

3

DTSFILE must be *recovered*, that is, DTSANALS with the RECOVER function must be run.

You can check for the return code using conditional JCL and make the execution of a subsequent job step dependent on the return code.

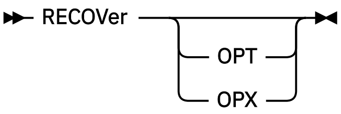
The ANALYZE command is used to sequentially read every chain of records on the VSE/ICCF library file. As each record is read, a bit map is checked to determine if the record has been accessed previously, the corresponding bit is set on and the next record is read. In addition to checking for previous reads, the routine validates the backward pointer. The forward pointer is assumed to be valid. If the backward pointer does not point to the record previously read, then it is changed to point to that record and the fact is logged on the printer. Even though this assumption is almost always a valid one, occasionally it may be necessary for a chain of pointers to be manually altered as described for the ALTER and DELETE commands.

The records are read in a specific order to aid in the resolution of duplicate pointers. This order is: system record, all user profile records, all broadcast records, library header records and all associated directory records and members. Common members are read only once but each occurrence of a directory record for a common member causes a check to validate that the member has been read and that the directory points to the first record of the member.

Analysis performs two corrective actions. First, any backward pointers which are not correct are corrected. Second, any chain of records (directory, members or free) which overlaps another chain is terminated at the record before where the overlap would occur. This is most common where member chains are intersecting the free chain because of an abnormal termination.

In each case the user is notified of the corrective action and the relative record number of the record involved. The user can, if desired, take further action at the end of the analysis phase.

RECOVER Command



OPT

Allows to make execution of the RECOVER command conditional. If OPT is specified, the command is executed only if DTSANALS recognizes that the VSE/ICCF library file was not closed properly during the last execution of VSE/ICCF or of a VSE/ICCF utility. However, to find out whether the VSE/ICCF library file has *really* lost its integrity, you must request an unconditional (that is: without the OPT operand) run of the ANALYZE command of DTSANALS.

OPX

If DTSFILE indicators signal that recovery is needed then this command is identical to 'RECOVER OPT'. If DTSFILE indicators show that no recovery is needed then DTSANALS searches all member directory records for update-in-progress flags, resets them and writes message K334I for which member the flag has been cleared. During CICSICCF startup this operand should only be used when there are frequently members found with UPIP flags on.

The program examines the bit map created during analysis to find unreferenced records. When one is found, the backward pointer is followed until a zero forward pointer is encountered or a record is read which has been read before. At this point, the first five records of the chain are printed to tell the user what chain of records has been found. Then a count is made of the records until the forward pointer is zero or until the forward pointer points to a record read previously. This count of records is printed and this chain is attached to the front of the free chain. This process is continued until all records have been accounted for.

In addition to the above checks, one other check is made. Because record additions are made from the area past the last record used (until this space is exhausted), it is possible that library members have used records from this area without the system record on disk reflecting this fact. In order to handle this situation, the program checks for record usage beyond this high file record number. If any usage has occurred, the program corrects the value in the system record and puts any unused records into the free chain.

Because the REORG command also performs the ANALYZE and RECOVER functions, it could be considered an analysis command. However, its effect is described in section “Reorganization Commands” on page 96.

Free Chain Interpretation

When recovery is complete and chains of records have been attached to the free chain, the user is faced with the problem of determining which chains are to be saved and which are to be left on the free chain. The following discussion is aimed at making that decision a little easier.

As records are added to the free chain as a result of deletes and member purges, their forward pointers are altered as needed to make them a part of the free chain. However, their backward pointers are not changed. This fact, together with the recovery technique of chasing backward pointers, may sometimes cause printing of numerous small subchains on the recovery report when actually there is only one long chain. This occurs only once after a chain is added to the free chain. After a recovery is complete, all backward pointers on the free chain have been corrected.

For this reason it is a good idea to periodically perform an analysis and recovery (unless BACKUP/RESTORE is run often) regardless of whether or not abnormal terminations have occurred. The analysis report is a most valuable aid in determining file conditions. If no error conditions are noted during analysis, recovery will only involve reading the free chain. If a member chain is terminated, the owner of that member should be consulted to be sure that vital records have not been lost. If a backward pointer has been corrected, it is a good practice to also check that member carefully because this could mean that the free chain has been attached to the member.

Another type of subchain which appears on the recovery report is the print or punch area chain. These usually contain data that makes them obviously print or punch output. Another aid for spotting print spool areas is that they are generally of the same length. That is, if there are six subchains, all 400 records long, they are probably print spool areas and can be left on the free chain.

A subchain with a large number of records indicated is probably a segment of the free chain that was lost when the free chain was terminated due to overlapping pointers. Check the analysis printout to be sure that the free chain was terminated. If the free chain was not terminated, it may be a large member which was purged.

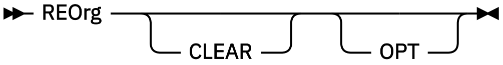
It is suggested that the installation keeps a dummy member name on one library which can be used to receive any chains of records uncovered during recovery having unknown ownership. That is, if a chain of records may be part of a member lost through an abnormal termination, it is a good idea to INSERT the records into this dummy member until the owner can be located. If it later turns out that they should be on the free chain, they can be deleted from this dummy member.

Reorganization Commands

Reorganization is never required but it can improve performance, especially when the available high file area is exhausted. Because the resource requirement for the REORG is in most cases the same as for BACKUP/RESTORE, it is recommended that BACKUP/RESTORE commands of DTSUTIL be used for reorganization. A regular reorganization of the DTSFILE improves performance in several ways:

- The records used for spool allocation are placed as close to each other as possible.
- The records acquired for input and additions to members are placed as close to each other as possible.
- The library directory records are placed adjacent to each other, so that the number of physical I/O operations required to do a directory lookup is reduced.

REORG Command



The REORG command takes the copy of the bit map produced at the end of the analysis phase and uses it to sequentially arrange the free chain records. When that function is complete, an attempt is made to reorganize the library directory records into consecutive records. The longest consecutive free area is used as the starting point for the directory reorganization. A directory chain is copied into this area. If all records fit, the library record is updated to point to the start of the new chain, the free chain is connected around this area, and the old directory records are connected to the end of the free chain. If the records do not fit, the reorganization of the directory records is terminated.

CLEAR
Causes the records in the free chain to be cleared to binary zeros.

OPT
Allows you to make execution of the REORG command conditional. If OPT is specified, the command is executed only if DTSANALS recognizes that the VSE/ICCF library file was not closed properly during the last execution of VSE/ICCF or of a VSE/ICCF utility. However, to find out whether the VSE/ICCF library file has *really* lost its integrity, you must request an unconditional (that is: without the OPT operand) run of the ANALYZE command of DTSANALS.

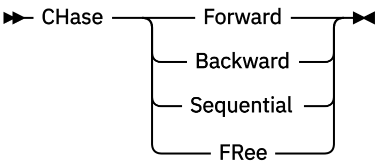
The three lines of statistics printed at the end of the reorganization step are for information only. They give you an idea of the arrangement of the free chain. The number of consecutive subchains is a count of the number of times that two or more records were physically adjacent in the free chain.

The average length of these subchains is obtained by dividing the total number of records in consecutive subchains by the number of subchains. The percentage figure is based on the ratio of the total number of records in consecutive subchains to the total number of records in the free chain. These figures may slightly misstate the actual situation on the file because of the technique used for finding consecutive subchains in the bit map.

Processing-Mode Commands

The mode of processing is the key to the record retrieval sequence. In addition to selecting whether the forward pointer, backward pointer, free chain pointer, or next sequential record number is to be used to get the next record, the user may put the program into the wait state, end all processing, change the form of input to card or console, or turn on the trace function.

CHASE Command



The CHASE command sets the source of the next record number to be processed by a printing, positioning or updating command. It is in effect until changed by another CHASE command, LIBRARY command, or RECORD command.

Forward
Causes the forward pointer in the current record to be used to get the next record for processing. Retrieval by forward pointer is in effect retrieval in the logical sequence of the file.

Backward
Causes the backward pointer in the current record to be used; that is, the previous logical record will be accessed.

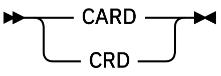
Sequential
Causes 1 to be added to the current record number, and that record will be the next record read. Thus, SEQUENTIAL is retrieval by physical record within the file rather than retrieval in a logical fashion.

FRee
Causes the first record of the free chain to be read replacing the current record. Any requests for more records will cause the forward pointer to be used.

FREE and SEQUENTIAL also reset any library number or member name previously given in a LIBRARY or MEMBER command.

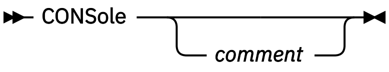
CHASE SEQUENTIAL is the initial mode of processing until the mode is changed, and is also the mode which is set after file analysis or file reorganization. Certain commands such as LIBRARY/MEMBER will automatically set FORWARD mode. The RECORD command will automatically set SEQUENTIAL mode.

CARD Command



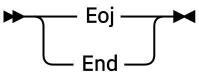
The CARD command causes all subsequent commands to be read from SYSIPT. If a command from SYSIPT is in error, the command will be displayed at the console, and the operator will have a chance to correct it. End-of-file on the reader forces an END command.

CONSOLE Command



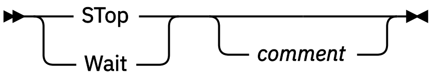
When read from the input stream, this command causes all subsequent commands to be read from the console. If a comment operand is specified, the comment or the ENTER DTSANALS COMMAND prompt line will appear.

EOJ Command



These commands indicate that processing is complete. All files are closed and the job is ended. (/* on SYSIPT indicates end-of-file and terminates DTSANALS.)

STOP Command



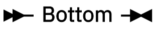
These commands cause DTSANALS to wait for an operator communication interrupt. They are usually used during console operations to free the console for messages from other partitions. MSG xx will cause the processing of commands to resume ('xx' is the partition where DTSANALS is running). When this command is entered from SYSIPT, the command along with any comment operand is logged on the console.

File-Positioning Commands

Positioning commands are those commands which alter the value of a file position pointer. They allow you to set the file position pointer by (1) specifying a record number, (2) specifying a displacement value from the current record (sequentially or along the forward or backward chains), or (3) by locating a particular string in the first 80 columns of the record. They also allow positioning to a member by name or to a library by number.

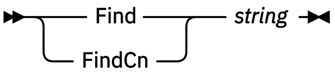
The FIND, BOTTOM, LOCATE, NEXT and UP commands function in a manner similar to their context editor counterparts.

BOTTOM Command



This command advances the pointer until an end-of-file condition is reached. This command is not valid in sequential mode. It *always* uses the forward pointer value as the next record number.

FIND Command



The Cn suffix may be used with this command. The FIND request causes a column-dependent comparison on the string within each line in the file. The compare begins on the next line from where the pointer is currently positioned and continues according to the previous CHASE command direction until a match occurs or until the end-of-file is reached. If 'string' is found, the pointer is positioned to the record in which 'string' is contained. The record is printed with its relative record number. If 'string' is not found, the pointer is positioned after the last line of the file.

Cn

Specifies the column from where the comparison is to start. If Cn is omitted, the comparison starts with column 1.

string

Is the character string on which the comparison is based. The string may be of any length but may not contain left or right parentheses, commas, or blanks.

FIND can be used to search for a specific line identifier in columns 73-80. One technique is to issue FIND with a C73 suffix. *Examples:*

1. FIND 90

```
request:      f 90
line found:    90   FORMAT (516)
```

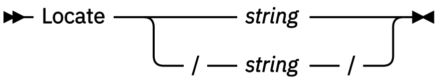
The FIND request searches for 90 in columns 1 and 2.

2. FC16 SUMX

```
request:     FC16 SUMX
line found:  LOOP   A   SUMX,X
```

The request searches for SUMX in columns 16-19.
The first line found is printed.

LOCATE Command



string

The slash (/) is the string-delimiter character. Its specification is optional and required only if blanks must be included. The string delimiter may be any non-blank character lower than 'A' in the sequence determined by the HEX-representation of the characters.

'string' is any group of characters to be searched for in the file.

LOCATE scans the characters of each record for the string specified. The scan begins on the next line following the line where the pointer is currently positioned and continues in the direction specified in the preceding CHASE command until the string is found or until the end-of-file is reached. If 'string' is located, the pointer is positioned at the line that contains it. If 'string' is not located, the pointer is positioned after the last line of the file.

The request is not column-dependent as all characters are scanned. *Examples:*

1. L FORMAT

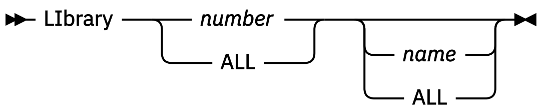
```
request:       loc   format
line located:  55   FORMAT ('DAILY AUDIT')
```

LOCATE searches all characters of each line for FORMAT.
The first line found is printed.

2. L /123   /

Locate searches each line for the specified character string.
The string delimiters were necessary in order to ensure that
the spaces were included in the string.

LIBRARY Command



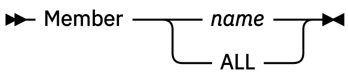
The LIBRARY command is used to indicate which library is to be searched for a member.

number
Is any (valid) library number, or ALL which indicates that every library is to be searched.

name
Is a member name, or ALL if all members within the library are to be processed by the following commands. This operand has the same function as a separate MEMBER command.

The LIBRARY command forces processing in FORWARD mode. See also "Note" under “MEMBER Command” on page 100 below.

MEMBER Command



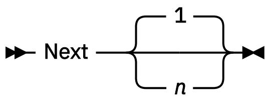
The MEMBER command indicates the name of the member to be processed.

name
Is the member name, or ALL which means that all members of the library specified previously will be logically concatenated for all functions except INSERT and DELETE.

The MEMBER command must only be given in the FORWARD mode of operation and must have been preceded at some time by a LIBRARY command.

Note: CHASE (FREE or SEQUENTIAL), ANALYZE, RECOVER, REORG, and RECORD all eliminate the values set by the LIBRARY and MEMBER commands.

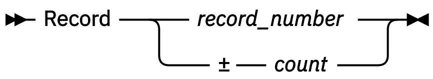
NEXT Command



This command advances the pointer in the file by 'n' lines.

n
If 'n' is not specified, a value of 1 is assumed and the pointer is advanced to the next line in the file. If end-of-file is reached before the pointer is advanced 'n' lines, the pointer is positioned after the last line. This command is not valid in sequential mode (use R +n). It *always* uses the forward pointer value as the next record number.

RECORD Command



This command sets the SEQUENTIAL mode of processing and retrieves the specified record.

record-number
Is an unsigned hexadecimal number from 1 to 8 digits specifying the relative record number within the VSE/ICCF file.

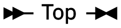
±count
If the count option is used, the operand must be a signed decimal number. This will reset the position pointer by the number of records specified.

Examples:

RECORD 2A43 Sets the file position pointer to
the 10819th record (relative to zero) of
the VSE/ICCF file.

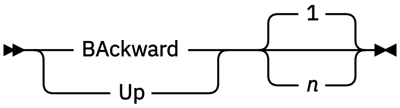
REC +6 If this is the next command, the pointer
will be set to the 10825th record.

TOP Command



This command repositions the pointer to the top of the library member (that is, to the null line in front of the user's first line in the file). This command is not valid in sequential mode. It always uses the backward pointer value as the next record number when searching for the top.

BACKWARD Command



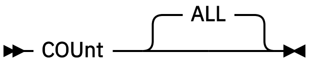
The UP or BACKWARD request repositions the pointer 'n' lines before the current line. If 'n' is not specified, a value of 1 is assumed, and the pointer is moved up to the previous line in the file. If 'n' is greater than the number of lines between the top of the file and the current line, the request functions as a TOP request. This command is not valid in sequential mode. It always uses the backward pointer value as the next record number.

Print- and Punch-Control Commands

These commands allow records from the VSE/ICCF library file to be printed in character or hexadecimal format with forward and backward pointers and record number. They allow such records to be printed in character format, first 80 columns only, or to be output on SYSPCH, again first 80 columns only. In addition, a count can be made without printing any records, head of forms may be forced, and the directory records of the libraries may be printed.

These commands always start at the current position of the file pointer, do not change the file position pointer, and always read the next record according to the current mode of processing.

COUNT Command



The COUNT command reads records beginning with the current record and increments a counter by 1 for each record read. When end-of-file or end-of-chain is encountered, the count is printed on SYSLST. ALL is the only operand. If omitted, it is assumed. The COUNT command is valid in any processing mode. To count the records in a given member, the user must be in FORWARD mode via the LIBRARY command.

DSERV Command

►► DServ ►◄

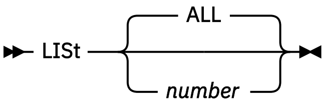
The DSERV command prints the contents of the directories of the VSE/ICCF library file. The directories to be printed are determined by the previously given LIBRARY command. All information contained in the directory is printed. All directory entries for a library are printed. (See the DSERV command of DTSUTIL for a more complete directory display.)

EJECT Command

►► EJect ►◄

The EJECT command causes a skip to channel 1 and the heading lines to be printed. This is useful if it is desired to start a printout at the top of a page.

LIST Command

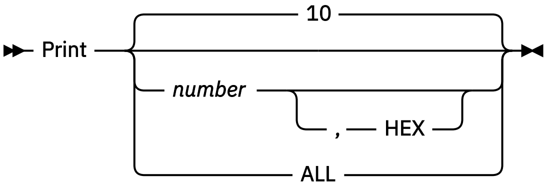


The LIST command causes the data portion of the record (first 80 columns) to be printed in character format.

number
Specifies the number of records to be printed. If it is omitted, ALL is assumed. ALL causes the printing to continue until end-of-file or end-of-chain has been reached.

This command may not be given in SEQUENTIAL mode.

PRINT Command



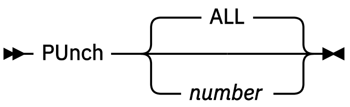
The PRINT command prints all 80 columns of the data portion of the record in character form plus the forward and backward pointers and the record number in hexadecimal.

number
Is the number of records to be printed. If 'number' is omitted, 10 records are printed.

HEX
Causes the specified number of records to be printed in character format, and immediately beneath each of the first 80 positions is the hexadecimal representation of that position. The HEX operand is only valid if 'number' is specified.

ALL
Causes records to be printed until the end-of-chain or end-of-file condition is reached.

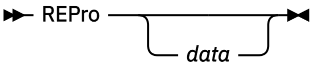
PUNCH Command



The PUNCH command causes the data portion of the record (first 80 columns) to be punched in character format.

number
Specifies the number of records to be punched. If the operand is omitted, ALL is assumed. ALL causes the punching to continue until the end-of-file or end-of-chain condition has been reached.

REPRO Command

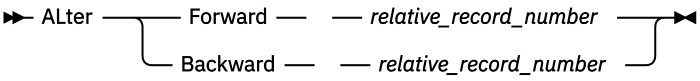


The REPRO command will punch the data supplied as the operand of the command into the punched output stream. If no operand is present, the entire card image which follows the command line will be written to the punch output stream. This command may be used to insert control cards between members being punched.

File-Updating Commands

The updating commands are those that change records on the VSE/ICCF library file. All of the commands except ALTER relate to a library and member, or all libraries or all members of a particular library. Therefore, the LIBRARY and MEMBER commands must have been issued previously to establish the file position. This group of commands allows records to be taken from the free chain and added to a member, records to be deleted from a member, forward and backward pointers to be changed, and flag bits to be reset in the directory records.

ALTER Command



The ALTER command is the only updating command which is not given in the FORWARD mode. It must be given in the SEQUENTIAL mode only. The RECORD command usually precedes the ALTER command in order to access the record to be altered and to set sequential mode.

Note: Be careful when you use the command; else you might destroy a file beyond recovery.

Forward
Indicates that the FORWARD pointer is to be altered. This operand may be abbreviated to one character.

Backward
Indicates that the BACKWARD pointer is to be altered. This operand may be abbreviated to one character.

relative-record-number
One to eight hexadecimal digits which will be the new value of the selected pointer.

The change is made to the current record. This command is most useful in terminating a chain by altering the FORWARD pointer of the last desired record to 0. The ALTER command should be followed by the RECOVER command, to ensure the integrity of the VSE/ICCF library file.

RESET Command

```
▶▶─ RESet ──  ── operands ─▶◀
```

The RESET command will turn off flag bits in the directory entries of whatever members have been specified through previous LIBRARY and MEMBER commands. The operands designate the particular flag bits to be reset.

UPD
Update-in-progress

CMN
Common member

PRV
Private member

FLG
Editor automatic change flagging

Two or more or all of these operands can be specified in any order. The specified operands must be separated from each other by at least one blank. At least one operand must be specified.

Examples:

```
LIB 3 MODULEA         Finds member MODULEA in library 3
RESET PRV UPD         Resets 'only private' flag for MODULEA
```

INSERT Command

```
▶▶─ INSert ──  ── first ──  ── last ─▶◀
```

The INSERT command is used to move records from the free chain into a member. The records are always inserted after the current record pointer location. After the insertion the file position pointer points to the last record inserted.

first last
Two numbers, each consisting of 1 to 8 hexadecimal digits. They represent the relative record numbers of two records on the free chain. The free chain is read until the record whose number matches the first operand is found. Reading of the free chain continues until a record is read whose number matches the last operand.

These two records, along with the chain of records between them, are attached (by pointer manipulation) to the record pointed to by the file position pointer. The free chain is reconnected so no gap exists. The INSERT command is used primarily after the RECOVER command to reattach chains of records which may have been detached due to system failure during certain edit or update functions.

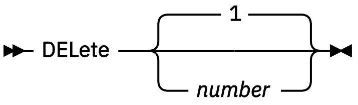
The INSERT command is interruptable during the search of the free chain, but not once the records have been located. The interrupt feature is useful in the event that the operator realizes that a match will never be found because of an input mistake.

Note: Be careful when using the INSERT command following a REORG. REORG manipulates the free chain records, and your reference to record numbers may be wrong.

Example:;

```
LIBRARY 6 MEMBRA
NEXT  27
INSERT 34AC 3B2A    (Insert records from free chain
                   after 27th record in MEMBRA)
```

DELETE Command



The DELETE command causes records from a member to be removed and to be placed into the free chain.

number
Is a decimal number indicating the number of records to be deleted. If the number of records to be deleted exceeds the number of records remaining in the member, the remainder of the member will be deleted and the next command will be read. Thus, it is possible to delete the remainder of a member by specifying a large number of records. If omitted, the number is assumed to be 1.

The current record pointer is counted as the first record to be deleted. The file position must have been established through the use of the LIBRARY and MEMBER commands. The DELETE command is not interruptible.

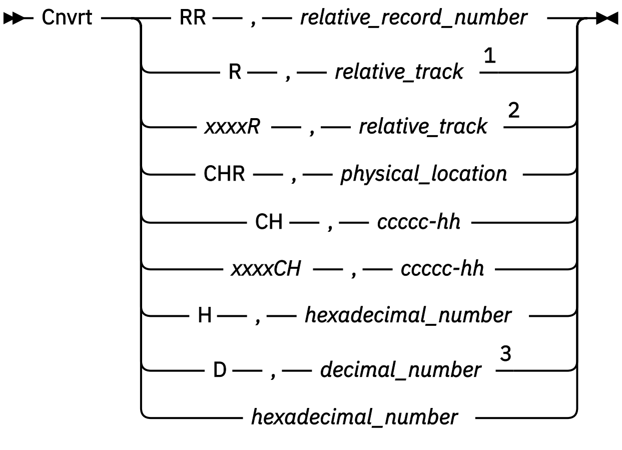
Note: If you want to transfer a group of records from one member to another, DELETE could be used to put the records into the free chain and INSERT could be used to put them into their new position.

Conversion Commands

The DTSANALS utility provides certain conversion routines. These routines help you diagnose problems involving the VSE/ICCF library file. For count-key-data (CKD) devices, they may be used to locate a record physically through the relative record number, find the relative record number through the physical location, convert a hexadecimal number to decimal, convert a decimal number to hexadecimal, find the relative track number from the cylinder and head, and find the cylinder and head numbers from the relative track numbers. For fixed-block (FBA) devices, they may be used to locate the physical block and the byte offset of a relative record into the physical block.

These commands are primarily to be used at the system console, since they always display the output on, and read the next command from that console.

CNVRT Command



Notes:

[1] Not allowed if VSE/ICCF library file is on a fixed block (FBA) disk.
[2] Not allowed if VSE/ICCF library file is on a fixed block (FBA) disk.
[3] Optionally with a sign.

The command converts a number from one format to another. The first operand indicates the format of the argument and directly indicates the format of the output. If the first operand is not one of the values listed, 'H' is assumed.

RR

Indicates conversion from relative record number to physical location. Physical location is either of the following:

- *For a CKD disk* — Programmer logical unit, cylinder number, head number, record number, logical record number within a block (sss-ccccc-hh-rrr-ll).
- *For a fixed block disk* — Programmer logical unit, physical block, and offset of a relative record into the physical block (sss-ppp-oo).

See the CHR operand for an explanation of the format. This command is used when a record's physical location needs to be determined from the relative record number. The argument is 1 to 8 hexadecimal digits.

R

Indicates conversion from relative track to cylinder and head, for the device type of the VSE/ICCF file. The second operand is a decimal number. It is not checked for validity relative to the device type. The second operand is divided by the number of heads per cylinder to give cylinder, and the remainder is the head number.

xxxxR

Indicates conversion from relative track to cylinder and head for the device you specify for xxxx. Your specification for xxxx is the IBM device type code for any supported CKD disk device. All other functions are as discussed for R, above.

CHR

Indicates conversion from a physical address in the VSE/ICCF file to a relative record number. The second operand, the device address, is in the format:

sss-ccccc-hh-rrr-ll For a CKD disk.
sss-ppp-oo For an FBA disk.

where:

sss
= Programmer logical unit number as specified in the EXTENT statement.

ccccc
= Cylinder number.

hh
= Head number.

rrr
= Physical record number on the track.

ll
= The logical record number in the block (relative to zero).

ppp
= Physical block number of an FBA device.

oo
= Offset of relative record into FBA physical block.

Each part of the operand must be separated from other parts of the operand by at least one non-numeric character. Each part of the operand may be up to four digits long, and no part may be omitted. The returned value is an eight-digit hexadecimal relative record number.

CH

Indicates conversion from cylinder and head to relative track number for the device type of the VSE/ICCF file.

You specify the second operand in two parts separated by at least one non-numeric character (for example: ccccc-hh), or as one part in the form cccchh.

VSE/ICCF:

1. Multiplies the cylinder number (ccccc), a number of up to four digits, by the number of heads per cylinder.
2. Adds the specified head number (hh), a number of one or two digits, to the result of the multiplication to get the requested relative track number.

No check is made for validity according to device characteristics.

xxxxCH

Indicates conversion from cylinder and head to relative track number for the device you specify for xxxx. Your specification for xxxx is the IBM device-type code for any supported CKD disk device. All other functions are as discussed for CH, above.

H

Indicates conversion from hexadecimal to decimal. The second operand is 1 to 8 hexadecimal characters. The returned value is an edited decimal number which corresponds to the hexadecimal.

D

Is used to convert from decimal to hexadecimal. The second operand is an optionally signed decimal integer not exceeding ± 2,147,483,647. The returned value is an 8-digit hexadecimal number.

Table 4. DTSANALS CNVRT: CKD Examples

Command:	CNVRT	FFFFFFFF	Requests decimal equivalent.
Result:	-1		Reply
Command:	C	CH,1-1	Relative track number.
Result:	16		If the DTSFILE is on a 3380.
Command:	C	CHR,10-326-1-4-0	SYS010, cyl. 326, head 1, record 4, logical record 0.
Result:	000001E4		Relative record number.
Command:	CNVRT	RR,34A	Requests the physical location of the record.
Result:	10 326 2 5 8		SYS010, cyl. 326, head 2, record 3, logical record 1.
Command:	C	3380CH,1-4	Give the relative track number of cyl. 001 head 4 on a 3380.
Result:	19		
Command:	C D, +58		Give the hexadecimal equivalent of 58.
Result:	0000003A		

Table 5. DTSANALS CNVRT: FBA Examples

Command:	C	CHR,10-804-88	SYS010, physical block, offset of the relative record into the physical block.
Result:	0000603A		Relative record number of VSE/ICCF library record on FBA.
Command:	CNVRT	RR,603A	Requests the physical location of the record.

Table 5. DTSANALS CNVRT: FBA Examples (continued)

Result	10 804 88		SYS010, physical block number 804 with an 88-byte offset into the physical block.

DTSANALS Utility Examples

The following examples assume that DLBL/EXTENT information for the DTSFILE is stored in the VSE standard label area.

Example 1: Display all directories for all libraries within the VSE/ICCF file.

```
// JOB DISPLAY DIRECTORIES
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSANALS
LIB ALL
DSERV
/*
/&
```

Example 2: Print two members from library 7 with chain pointers, then print three members from library 13 without chain pointers.

```
// JOB DISPLAY MEMBERS
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSANALS
LIB 7
MEMBER RORTPRA
PRINT 9999
MEMBER RORTPRB
PRINT 9999
EJECT
LIB 13 FORTDTA
LIST
MEMB   ORTDTB
LIST
MEMB   ORTDTC
LIST
/*
/&
```

Example 3 : Print all members in library 10 in character and hexadecimal with chain pointers.

```
// JOB PRINT
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSANALS
LIB 10
MEMBER ALL
PRINT 9999,HEX
/*
/&
```

Example 4: Reset the update-in-progress flag bit for a specific library member (JAC01).

```
// JOB RESET
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSANALS
LIB 14
MEMB JAC01
RESET UPD
/*
/&
```

Example 5: Place data records from the free chain into a library member called DUMPDATA. The beginning and ending relative record numbers of the data in the free chain (24AC,1F40) were determined by printing the free chain as in the preceding example.

```
// JOB RECOVER DATA
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSANALS
LIB 4
MEMBER DUMPDATA
BOTTOM                  (to place the data at the end of the member)
INSERT 24AC,1F40
TOP
EJECT
LIST                    (to print the member with the new data)
/*
/&
```

Example 6: Locate the string 00007100 in the free chain and print the corresponding record number. Then locate the record that contains the string 00019700 and print its record number. Then go to the system console so the operator can enter commands to reinsert the data into the member TESTMEMB, 6 statements behind the record that contains the string START.

```
// JOB ANALS
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSANALS
CHASE FREE
LOCATE /00007100/
PRINT 1                 (record found at location 237A)
LOCATE /00019700/
PRINT 1                 (ending record at 42FC)
CONSOLE
LIB 12,TESTMEMB         (entered at console)
FIND START              (entered at console)
NEXT 6                  (entered at console)
INSERT 237A,42FC        (entered at console)
EOJ                     (entered at console)
```

Example 7: Execute DTSANALS from the card reader and read control information from the system console.

```
// JOB ANALS
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSANALS
CONSOLE
/*
/&
```

Data-Change Audit (DTSAUDIT) Utility Program

Data processing installations need to periodically audit changes to the VSE/ICCF library file. The DTSAUDIT utility allows these users to obtain reports concerning additions, changes and deletions to the VSE/ICCF library file.

The DTSAUDIT utility program has many auditing and tracing capabilities. Some of the more important functions are the following:

- The ability to scan the entire file, to scan only a given library, to scan only certain libraries, to scan a given member or to scan a group of members for changes.
- The ability to produce reports giving only the changes to the file or reports containing lists of the entire library member with the alterations (additions, deletions, changes) indicated where they occur within the member.
- The ability to select one or more of the following options for given libraries or given library members:
  - The *physical scan* option

    It indicates all physical additions and deletions to the data being scanned. This scan also indicates whether a member is entirely new. All physical scanning is based on a checkpoint concept which will be described presently. A physical type of scan may be made on any library member or library to indicate additions and deletions.
  - The *logical scan* option

  It indicates all logical additions, changes and replacements to the file based on the VSE/ICCF editor change flagging option. Any record flagged by the editor as a change, replacement or addition is indicated in the change report. This type of scan may be made on any library member but only has meaning for those members which have been changed with the editor change flagging option set on.

  - The *sequence number scan* option

    It scans for sequence number alterations in order to indicate additions or deletions to the data being scanned. This type of scan would only be appropriate for members which have imbedded sequence numbers within the data.

- The ability to locate which library a given member occurs within. Also the ability to obtain scans or listings of a given member name in all libraries in which the given member name occurs.
- The ability to obtain offline (batch) listings of members which are in compressed format within the library.
- The ability to execute the utility in either a batch or interactive partition while VSE/ICCF is running. If executing in an interactive partition, the VSE/ICCF administrator can prepare audit reports while VSE/ICCF is running. VSE/ICCF users (other than the VSE/ICCF administrator) may also use DTSAUDIT while VSE/ICCF is running; however, they may only have access to members to which they would normally have access. Thus, VSE/ICCF users (other than the VSE/ICCF administrator) may use DTSAUDIT to obtain reports of changes to modules for which they are allowed to make changes.

The primary purpose of the DTSAUDIT program is to allow an installation to monitor additions, deletions and changes to data within the library file for reliability, or access control purposes. To help the installation obtain the required information, three different options for viewing the data are available. The options are the physical scan option, the logical scan option and the sequence number option. These options may be used singly or in combination on a single member or library or on multiple members or libraries.

Physical Scan

The physical scan option will produce reports containing all physical additions and deletions to the data being scanned since the last checkpoint. The checkpoint concept is important only with the physical scan option. A checkpoint in terms of the DTSAUDIT program is the execution of the backup and restore function of the DTSUTIL utility against the VSE/ICCF library file.

At the time of the checkpoint (restore) the high end of the data within the library file is indicated within the system record of the library file. Also at the time of the restore, all records which are logically contiguous are made physically contiguous. During the DTSAUDIT physical scan of a member, any disruption of the physical sequence of the file compared to the logical sequence of the file indicates the presence of a deletion or addition to the file:

- Any time the logical sequence of records skips across physical records in the checkpointed area of the file, a deletion is indicated. If the deleted records are still available, they will be printed in the report. If not available, an indication will be made as to how many records have been deleted.
- Any time the logical sequence of records skips to the newly allocated area of the file (beyond the checkpoint indication), one or more additions are indicated. These additions are printed.

  If the entire member lies beyond the checkpoint area, the entire member is indicated as new. The physical scan option does not apply to compressed library members; however, the logical and sequence number options described below do apply to compressed members.

In order to use the physical scan option, you should:

1. Back up and restore the VSE/ICCF library file on some periodic basis. This period may be anything from one day to perhaps two weeks. This backup/restore will be called the checkpoint.
2. Ensure that the VSE/ICCF library file is large enough so that, within the checkpoint period, all new records are allocated in the high file area rather than in the free chain. (Free chain records are used only after the high file area is used up.) This is important if you want to actually see the physical

records deleted rather than just get an indication of the point of deletion and the number of records deleted.

3. Each day within the checkpoint period, run the DTSAUDIT utility to obtain a report of changes to the file since the start of the checkpoint period. DTSAUDIT should be run immediately prior to doing the checkpoint (backup/restore) so that a final report of all cumulative changes is available.

Logical Scan

The logical scan option produces reports containing all logical changes to the file based on the editor flagging option. If the editor flagging capability is used within VSE/ICCF to edit certain (or all) library members, then each altered record will be flagged as to the date and type of the change and the user making the change. The logical scan option of the DTSAUDIT utility causes members to be scanned for the editor flag. When a flagged record is found, it is printed on the report. The report indicates the date of the change and the user ID of the user who made the change. In addition, the type of the change (addition, data change, line replacement) is also indicated.

In order to use the logical scan option, the installation should:

1. Set the editor flag option on for all members which are to be edited with this option. This option can be set on by using the /PROTECT command with the flag option, or during editing of a member via the FLAG command. The flagging columns are defined in the EDFLAG tailoring option of the DTSOPTNS macro.
2. Run the DTSAUDIT program using the logical scan option to print all records which are found to have been flagged by the editor. The DTSAUDIT program can be made to scan all members within a library or only those members which have had the editor flag option set on via the /PROTECT command.

The Sequence Number Scan

The sequence number scan option produces reports containing any deletions or additions to the file as indicated by alterations in the normal sequence number progression within the data being scanned. This option is meaningful for library members which contain imbedded sequence numbers within the individual records themselves.

The sequence number scan is based on an increment value associated with the last resequencing of the member. For example, if the member was resequenced with an increment of 100 and if a certain progression of sequence numbers within the file was 002600, 002610, 002620, 002700 and 002800 it would indicate lines 002610 and 002620 as additions to the file. Similarly if the progression was 005100, 005200, 005500 and 005600, it would indicate that lines 005300 and 005400 had been deleted.

The only requirement for use of the sequence number checking option is that the data being scanned must contain imbedded sequence numbers and that these sequence numbers were generated using a fixed increment value at the time of the last resequence.

DTSAUDIT Commands

The DTSAUDIT utility recognizes several commands. The primary command is the PRINT command. The PRINT command requests an actual DTSAUDIT scan of the library file. All commands must begin in column 1 of the input control card image and must be wholly contained within a single card image. All commands may be abbreviated down to the minimum form as described below. Command operands must be separated from each other by commas or spaces. Multiple commas or spaces are treated as single commas or spaces. All operands may be abbreviated down to the point where they become non-unique among the set of all operands.

DTSAUDIT commands may be entered from either SYSIPT or from the console (SYSLOG). If DTSAUDIT is initiated from the console, command input will be from the console. If DTSAUDIT is initiated from SYSRDR, command input will be from SYSIPT. SYSIPT may also be assigned to disk or tape if desired.

The actual commands available are:

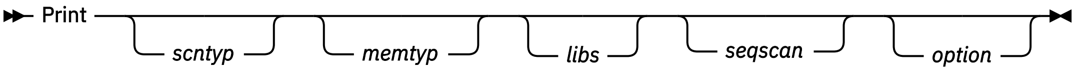
- PRINT – The PRINT command requests a specific type of scan of the VSE/ICCF library file. This command controls whether physical, logical or sequence number options (or a combination of two or three options) will be performed against the requested data or whether the data are to be printed without any scan being performed on them (NEITHER). This command also specifies whether a single member, an entire library or the whole library file will be scanned.

- OPTION – The OPTION command allows you to set several of the DTSAUDIT program options on a global basis. That is, once an option is set via the OPTION command, it will remain in effect for all subsequent PRINT commands until reset by another OPTION command. Thus, when an option is set this way, it need not be specified again on each subsequent PRINT command.
- CARD – The CARD command has no operands. It may be used to set the control statement input device from the console to the card reader (SYSIPT).
- CONSOLE – The CONSOLE command has no operands. It may be used in the SYSIPT job stream to set the control statement input device from the 'card reader' (SYSIPT) to the console.
- END – The END command is used to terminate control statement input. It would mainly be used when entering commands from the console since the /* would normally terminate commands entered via SYSIPT.

Note:

1. Do not use the SIZE operand of the // EXEC statement.
2. Long running commands may be interrupted from the system console via the VSE operator command MSG BG or MSG Fn. If a command is interrupted in this manner, a new DTSAUDIT command will be requested from the system console. The console operator may enter one or more commands in this manner. If more commands are waiting to be processed in the SYSIN job stream, the console operator may enter the CARD command to return command reading to SYSIN.

PRINT Command



scntyp
Is the physical/logical scan type to be performed. The possible options are 'BOTH', 'LOGICAL', 'PHYSICAL' or 'NEITHER' indicating that both physical and logical scans are to be performed or either a logical or a physical scan will be performed or that neither physical nor logical scans will be performed. The default is 'BOTH' if this operand is not specified.

memtyp
Indicates which members within the libraries specified in the 'libs' operand are to be scanned. The possible operands are 'ALL', 'FLAGGED', 'MEMBER xxxxxxxx PASSWORD yyyy'. 'ALL' indicates that all members within the library(s) indicated via the 'libs' operand are to be scanned. The 'FLAGGED' operand indicates that only those members which have been set for editor change flagging (via the /PROTECT command or the FLAG editor command) are to be scanned. The 'MEMBER' operand specifies that only the member whose name is specified (xxxxxxxx) is to be scanned. If running in an interactive partition and you are not the VSE/ICCF administrator, the 'password' portion of this operand must also be specified provided the member requires a password. 'ALL' is the default.

libs
May be specified as 'ALL' or 'LIBRARY n'. 'ALL' is the default which indicates that the member or members specified in the 'memtyp' operand will be scanned in all libraries within the VSE/ICCF library file. If 'LIBRARY n' is specified, only library 'n' will be scanned.

seqscan
Indicates that a sequence number option type of scan is desired on the member(s) specified. This option may be used by itself or with either the physical/logical or both options. This operand is specified as 'SEQUENCE n1 n2 n3'. 'n1' is the starting column number in the record where the sequence number field occurs, 'n2' is the number of columns in the sequence number field and 'n3' is the increment used during the last resequence of the member.

option

May be specified as one of the following:

EJECT

The operand indicates that the printer should be skipped to head of form before printing the output report for each member.

FULLPRINT

The operand indicates that all lines within the members being scanned should be printed rather than just the default, which is printing only the added, changed or deleted lines.

RESET

The operand causes the editor flagged records in any member being scanned to be reset. The next time the DTSAUDIT program is run, the previously flagged records are therefore no longer indicated as changed. RESET causes the editor flag area within a flagged record to be overlaid with four asterisks and the month and day of the RESET.

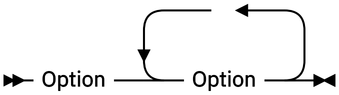
The option is effective only when DTSAUDIT runs in a VSE batch partition and VSE/ICCF is not up and running.

Flags are not reset for a member that is compressed.

SORTED

The operand indicates that library directories are to be sorted by member name before processing.

OPTION Command



option

Are one or more options which are to be set for all subsequent print commands. Once an option has been set via the 'OPTION' command, it remains in effect for all subsequent print commands so that the particular option need not be specified on each 'PRINT' command. All previous 'OPTION' command settings are reset each time a new 'OPTION' command is read. The possible command options are explained below:

EJECT

Indicates that a skip to the head of the form is to be performed prior to printing the change report for each member processed by the DTSAUDIT utility.

FULLPRINT

Causes not only the changes to the member to be printed but the entire member. Any changed lines will be flagged so that they will be distinguishable from the unchanged lines.

NOEJECT

Indicates that a skip to head of form is not to be performed for each new 'PRINT' command. Normally, each new DTSAUDIT 'PRINT' command read from the command input device will cause the printer to advance to the head of the form. This option causes this skip to head of form for each print command to be bypassed.

RESET

Indicates that all members logically scanned which contain editor flagged records will have the editor flag area reset or overlayed within the records so the records will not appear as changed on subsequent DTSAUDIT reports. Flags are not reset for a member that is compressed.

SORTED

Indicates that all library directories processed by the DTSAUDIT utility are to be sorted into alphabetic sequence by member name prior to processing.

Interpreting the Printout

The printed output from the DTSAUDIT utility program is for the most part self-explanatory. However, the notations indicated at the right of the printed lines may need some explanation.

Any line which is found to be flagged according to the options requested (physical, logical or sequence scan) will have an indicator written in columns 83 through 120 of the printed line.

The heading for this area is:

```
  **PHYS** ***LOGICAL*** **SEQ SCAN**
```

Any addition or deletion to the file indicated via the physical scan option will be noted under the '**PHYS**' portion of the heading. Any change to the file indicated via the logical scan option will be noted under the '**LOGICAL**' portion of the heading. Any addition or deletion to the file indicated via the sequence scan option will be noted under the '**SEQ SCAN**' portion of the heading.

Physical Scan Indicators ('**PHYS**')

Indicator = ADD
The record was added to the member following the last file checkpoint.

Indicator = ADD FREE
The record was added to the member following the last file checkpoint. The new record itself was allocated from the imbedded free chain because the unused area at the end of the library file was exhausted.

Indicator = ERROR-n
During the physical scan of a member, a forward pointer was encountered that pointed to a record with an address lower than the current record and with an address beyond the high end after the last restore.

n = 1
The chain points outside the member (no free chain exists from which the record with the low address could have been added).

n = 2
The chain may be incorrect. You have to check if the forward pointer points to the next member record or outside the member.

Indicator = ********
This indicates the record immediately prior to one or more deleted records. It is displayed only as an indication of the point in the file at which records have been deleted.

Indicator = ADD*****
This indicator has the same meaning as the 'all asterisks' above except that the record immediately preceding the deleted records was also an addition to the file; that is, it was added to the file following the last file checkpoint.

Logical Scan Indicators ('**LOGICAL**')

The logical scan indicators are divided into three parts as follows:

```
  type     date     user

where: type = The type of the change.
       date = The date of the change (in the form mm/dd or dd/mm
              depending on the format of the date defined to the
              system).
       user = The identification of the user who made the change.
```

Indicator = ADD
The record has been added to the member following the last reset of the editor flags.

Indicator = CNG
The indicated record has been changed (via one of the editor commands which allows data within a record to be altered) following the last reset of the editor flags.

Indicator = REP
The indicated record has been completely replaced (or changed via the full-screen editor) within the member following the last reset of the editor flags.

Indicator = NEW
The indicated record has been added to the file or replaced in the file via the 'nn' editor command in conjunction with linemode editing.

Sequence Scan Indicators ('**SEQ SCAN**')

Indicator = NEW SEQ
The indicated record contains a sequence number which has been inserted into the member following the last resequencing of the member.

Indicator = DELETED
The sequence numbers in the range indicated have been removed from the member following the last resequencing of the member (assuming the resequencing was done according to the specifications indicated in the SEQUENCE operand).

Indicator = ***BAD SEQ***
The member is out of sequence according to the SEQUENCE operand specifications. Only 30 of these indicators are printed for a given member.

Audit Examples

The following examples illustrate various uses of the DTSAUDIT utility. They show only the DTSAUDIT control commands. It is assumed that the commands are surrounded by the appropriate VSE JCL if running in a VSE batch partition. DLBL/EXTENT information for the DTSFILE is assumed to be stored in the standard label area.

You may use the following job stream as a model:

```
// JOB DTSAUDIT - PREPARE FILE AUDIT REPORTS
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSAUDIT
   ... ... ...      | Commands go here
/*
/&
```

Example 1: Scan all members in all libraries performing both a physical scan and a logical scan. Submit either of the commands below.

```
  PRINT
  PRINT BOTH ALL ALL
```

Example 2: Scan all members in library 14 performing a physical scan, a logical scan and a sequence number scan. Process the members in alphabetic sequence by member name.

```
  PRINT LIB 14 SEQ 73 6 100 SORTED
```

Example 3: Scan all libraries for members which have the editor change flagging option indicated in the library directory entries. Perform both a physical and a logical scan of these members.

```
  PRINT FLAGGED
```

Example 4: Scan all libraries for all occurrences of a member named 'XFER'. Perform a physical, logical and sequence number scan on these members.

```
  PRINT MEM XFER SEQ 1 6 100
```

Example 5: Print a listing of the member called 'MYTEST' from library 5. Do not perform any of the scanning options.

```
PRINT NEITHER FULLPRINT MEM MYTEST LIB 5
```

Example 6: Perform a logical scan only on the member called 'VSMCOB' in any library in which the named member occurs. Print all records within the member.

```
PRINT LOGICAL MEMBER VSMCOB FULL
```

Example 7: Print a listing of all members in library 2. Make sure that the members are listed in alphabetic sequence by member name. Do not perform any of the scanning options. Skip to head of form at the beginning of each listed member.

```
PRINT NEITHER LIBRARY 2 FULLPRINT SORTED EJECT
```

Example 8: Determine in which library or libraries the member named 'ADRSLST' occurs.

```
PRINT NEITHER MEMBER ADRSLST
```

Example 9: Do a physical, logical and sequence number scan on libraries 9, 11, and 15. Begin the printed change report for each member on a new page. Also have the reports prepared in alphabetic sequence by member name.

```
OPTION EJECT SORTED
PRINT LIB 9 SEQ 73 6 100
PRINT LIB 11 SEQ 73 6 100
PRINT LIB 15 SEQ 73 6 100
```

Example 10: Perform miscellaneous scans on various members within the library file.

```
PRINT LIB 1 MEM ASMPRGA SEQ 73 6 100
PRINT LIB 1 MEM ASMPRGB SEQ 73 6 100
PRINT LIB 1 MEM ASMPRGC SEQ 73 6 100
PRINT LIB 4 MEM COBPRGA SEQ 1 6 100
PRINT LIB 4 MEM COBPRGB SEQ 1 6 100
PRINT LIB 6 MEM MYPROC PHYSICAL
```

Batch (DTSBATCH) Utility Program

The DTSBATCH utility can be used to enter VSE/ICCF system commands and editor commands in an offline manner via SYSIPT or SYSLOG.

The commands thus entered are processed against the VSE/ICCF library just as if they had been entered at the terminal; for exceptions see below, prior to "Notes®". If DTSBATCH is initiated from SYSRDR, the input commands are read from SYSIPT, and the command responses are directed to SYSLST. If DTSBATCH is initiated from SYSLOG, the input commands are read from SYSLOG and output is directed to SYSLOG thus simulating terminal interaction. If an output operation is in progress on SYSLOG, it may be terminated by entering the attention routine command

MSG nn

Note: DTSBATCH can not be executed in an interactive partition.

If the DTSBATCH utility is run while VSE/ICCF is active, no writes are performed against the VSE/ICCF library file. This means that while VSE/ICCF is active, the DTSBATCH utility should only be used for list, display or punch type functions. Write functions can be performed when VSE/ICCF is not active or when the VSE/ICCF library file has been disconnected. To perform functions similar to DTSBATCH in an interactive partition, DTSPROCS must be used instead. Refer to " /LOAD DTSPROCS" in the VSE/ICCF User's Guide.

The primary use of this utility undoubtedly is the listing or punching of library members, the listing of directory entries, the entering of data members to the library, and the updating or editing of those members. However, also any other command function may be performed; exceptions are: /

EXEC, /ENDRUN, /MSG, /RUN, /SEND, VSE/POWER interface commands like /LISTP and /CTLP, and the full-screen editor commands.

Note:

1. Logging on under a certain userid via DTSBATCH does not prohibit logging on to VSE/ICCF with the same userid at the same time.
2. When running DTSBATCH under VSE/POWER, input from SYSIPT may not contain VSE/POWER JECL.

Control Input

The control input for the DTSBATCH utility are card images of any foreground commands or data, which the user wishes to apply to the VSE/ICCF library file.

See the VSE/ICCF User's Guide for individual command formats.

UPSI bits 3 and 4 are used to control punching of library members. If '**// UPSI XXX10**' is specified, any time a /LIST, /DISPLAY or /LIBRARY command is encountered, the output is directed to the card punch, as well as to the printer. When '**// UPSI XXX11**' is specified, the normally listed output is directed to the card punch, and printing is suppressed.

The /LOGON command must be the first command read, otherwise DTSBATCH is terminated. The second input line read must be the logon password. The last command read should be the /LOGOFF command. Multiple /LOGON/LOGOFF groups may be specified in a single execution of the utility; however, each /LOGOFF prior to a /LOGON should be followed by a blank card. The last command read should be '**/***'.

Special Disk or Tape Input

Under DTSBATCH, a special form of the /INSERT system command and the GETFILE editor command are available. If the member name specified on either of these commands is $$FILE, the data inserted into the input area or the member being edited is from the tape or disk file called DTSBTCH.

Note: For compatibility reasons, the ETSS II special names are still accepted but the preferred name is $$FILE.

The DLBL/EXTENT for DTSBTCH and the assignments are required for disk or tape (see the example 3 below). If the input is from tape, the tape must be assigned to SYS004, have standard labels, and must be positioned to the input file via job control MTC commands.

The input file may be blocked or unblocked 80 character records. The maximum block size is 4000. Unblocked 81 character records are also acceptable causing the first byte of the record to be ignored.

DTSBATCH Examples

The following examples illustrate various uses of the DTSBATCH utility. In these examples, the blank line behind the password represents a blank card if input is from SYSIPT, and a 'press ENTER' if input is from the console or terminal. (DLBL/EXTENT information for the DTSFILE is not shown because it is assumed to be stored in the standard label information area.)

Example 1: Add a member to the library associated with user identification USRX. The member added is called TESTPROG and it is public data.

```
// JOB    DTSBATCH
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC   DTSBATCH
/LOGON USRX                  (userid)
PASWDX                       (password)
                             (blank line)
/INPUT                       (enter input mode)
 ... ... ...                 (member data)
/END                         (end of data)
/SAVE TESTPROG PUBL          (save data in library)
/LOGOFF
/*
/&
```

Example 2: Update an existing member in the library using the context editor. Change all occurrences of the character string 2314 to 3380 in the member named SAMPROG which is password protected.

```
 ... ... ...                 JCL as above
/LOGON USBA
AWL$$A
                             a blank line
/EDIT SAMPPROG BA$$          enter edit mode
CHANGE /2314/3380/ G *       change all occurrences
QUIT
/LOGOFF
```

Example 3: Add a member to the library where the member currently exists as a blocked or unblocked card image file on tape.

```
// JOB DTSBATCH
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// TLBL DTSBTCH (required)
// ASSGN SYS004,TAPE
// MTC REW,SYS004
// EXEC DTSBATCH
/LOGON USXX
AXXPAS
                            a blank line
/INPUT
/INSERT $$FILE
/SAVE   MEMBX
/LOGOFF
/*
/&
```

Example 4: Insert a file of blocked or unblocked card images from disk into a member called JACFIL within the VSE/ICCF library.

```
// JOB DTSBATCH
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// DLBL DTSBTCH,'SPEC.DATA.FILE'
// EXTENT SYS020,WRKFIL,1,0,1200,300
// ASSGN SYS020,DISK,VOL=WRKFIL,SHR
// EXEC DTSBATCH
/LOGON USRA
USRAXX
                                (blank line)
/EDIT JACFIL
LOCATE STARTX
GETFILE $$FILE
QUIT
/LOGOFF
/*
/&
```

Example 5: Punch a member named COGOPROG from the library and write the punched output to tape.

```
// JOB    PUNCH
// UPSI 00010
// ASSGN SYSPCH,280
// EXEC PROC=DTRICCF         (assign VSE/ICCF library file)
// EXEC DTSBATCH
/LOGON JACA
PASJAC
                              (blank line)
/LIST COGOPROG
/LOGOFF
/*
// CLOSE SYSPCH,cuu
// MTC WTM,SYSPCH
// MTC REW,SYSPCH
/&
```

Dump-Formatting (DTSFDUMP) Utility Program

The DTSFDUMP program is used to interpret and format VSE/ICCF tables and to display the formatted output on SYSLST.

The primary purpose of DTSFDUMP is to serve as debugging aid when a problem seems to be caused by a VSE/ICCF program. Its use, therefore, requires a fairly good knowledge if VSE/ICCF internals.

Contrast this program with the VSE/ICCF DUMP command(s), which have to do with user program storage. Refer to "Dump Commands" in the VSE/ICCF User's Guide.

DTSFDUMP runs either in a dedicated VSE partition or in a VSE/ICCF interactive partition. Its functions can also be invoked by the VSE/ICCF operator command /FDUMP. See "/FDUMP Command" on page 72.

DTSFDUMP Running in a VSE Partition

When running the DTSFDUMP utility in a VSE partition, you can set different modes of operation by UPSI bit settings:

- UPSI bit 0 is not set (// UPSI 0)

  For input DTSFDUMP expects a tape containing a VSE dump which was created via the attention routine command

  ```
  DUMP xx,cuu
  ```

  'xx' indicates the partition identifier, 'cuu' a tape drive where an unlabeled scratch tape is mounted. The tape must be assigned to SYS007.

  DTSFDUMP processes the tape as follows: it scans the tape for the identifier 'ICCF'. When this identifier has been found, tables are formatted and printed. A blank line is inserted at the start of each control block that was found. This way tables are easier to locate in the dump. The actual and relative addresses of all formatted tables are printed to make them easier to find in the assembler listings of VSE/ICCF. When the ICCF identifier is not found on the tape (EOF is found), an error message is printed and the job is terminated.

- UPSI bit 0 is set (// UPSI 1)

  A test is made to see whether VSE/ICCF is running (in another partition). If this is the case, DTSFDUMP writes a dump of that CICS/ICCF partition on SYSLST. The output has the same format as when produced from tape. This dump does not *seize* the system, which means that data in the tables and the final dump may differ. (The DUMP command mentioned above, however, does obtain a seized-system dump.)

  The DTSFDUMP partition must be located in the same address space as the CICS/ICCF partition. If the DTSFDUMP partition and the CICS/ICCF partition are not in the same address space or if VSE/ICCF is not running, an error message is printed and the job is terminated. Use of the /FDUMP command would be helpful in this kind of situation.

- UPSI bit 1 is set (// UPSI X1)

  Only the VSE/ICCF tables (the formatted parts of the dump) are printed. The storage dump is not printed.

- UPSI bit 3 is set (// UPSI XXX1)

  This setting requests a scan for a specified character string in the running CICS/ICCF partition or in the dumped partition on tape, depending on the setting of UPSI bit 0.

  You are prompted by a message to enter the scan start and stop addresses. If no address is entered, the scan is made for the entire partition. After the next message you may enter the scan string in hexadecimal (two characters per byte) or in character format (one character per byte). For character format, the scan string must be enclosed in quotes. An included quote in the string must be entered twice. The maximum length of the scan argument is 16 bytes. If the scan argument is entered in hexadecimal format, each halfbyte which is entered as 'X' (for example 47XXBX) may have any bit configuration in the dump (see the description on the following pages). Each match found is printed on SYSLST.

By specifying PARM='nnnn' in the EXEC statement, you can request a number of *forward space files* on the tape (up to a maximum of 9999). This function can be used if more than one dump is on the input tape.

DTSFDUMP Running in a VSE/ICCF Interactive Partition

When DTSFDUMP is run in a VSE/ICCF interactive partition, you can display various VSE/ICCF tables and display or alter areas in the supervisor or in VSE/ICCF. You may also scan within the CICS/ICCF partition. Figure 17 on page 120 shows some typical output from DTSFDUMP when running in an interactive partition. It is the first display after you invoked the program (by entering $DTSFDUMP).

```
  WORK     01    MRP      01    FINAL    01    CSA      01    RQE      01
  PICB     05    TCQ      06    MNECB    01    HIECB    01    DIFO     00
  EXPICB   05    LABINFO  00    AUXP1    01    AUXP2    01    AUXP3    01
  AUXP4    01    AUXP5    01    AUXP6    01    AUXP7    01    MFH      01
  MFCT     01    EXTENTTB 01    BUFTABLE 20    FILEBUFF 20    BUFF     20
  TAS      01    FSEP     00    FSCR     00    FSED     00    CPCCB    01
  SPL      01    XPCCB    01              *
          ENTER TABLE NAME AND NUMBER OF ENTRY
          OR EOJ FOR EXIT OR DD HEXADDR OR SA HEXADDR OR SCAN OR SCANF
```

Figure 17. Example of Output from DTSFDUMP

As the last two lines indicate, you have several 'commands' at your disposal:

table name
DD
SA
SCAN/SCANF
EOJ

Entering the table name and a number (of up to the value shown after the table name in the above display) formats the current content of a particular table entry at your terminal. For example, to display the sixth TCQ entry you have to enter:

```
TCQ 06
```

If you do not enter a number after the table name, the first entry is shown. Each table name may be abbreviated to at least one character. Note that the table names, as arranged in the above display, will be scanned from left to right and top to bottom (the scanning sequence is also reflected in the figure below). So if, for example, you enter only 'T', the TCQ entry will be displayed. But if you enter 'TTC', the TCCAS will be displayed.

Table 5 gives a list of the VSE/ICCF tables and their purpose.

Table 6. VSE/ICCF Tables and their Purposes

WORK	Common pointer area
MRP	Main routine pointer
FINAL	OPCM save area and final routine
CSA	Main Task Common System Area including DTSIGEN
RQE	Request Queue Entry
PICB	Interactive Partition Information Block
TCQ	Task Control Queue Block
MNECB	Main Task ECB
HIECB	Timer Interrupt ECB
DIFO	Dynamic Disk Information Block
EXPICB	Interactive Partition Information Block Extension
LABINFO	Label Information Block

Table 6. VSE/ICCF Tables and their Purposes (continued)

AUXP1	Save Area in Interactive Partition Part One
AUXP2	Save Area in Interactive Partition Part Two
AUXP3	Save Area in Interactive Partition Part Three
AUXP4	Save Area in Interactive Partition Part Four
AUXP5	Save Area in Interactive Partition Part Five
AUXP6	Save Area in Interactive Partition Part Six
AUXP7	Save Area in Interactive Partition Part Seven
MFH	Main File Handler Pointer
MFCT	DTSFILE Control Table
EXTENTTB	DTSFILE Extent Table and Set Sector Value Table
BUFTABLE	Buffer Table Entry
FILEBUFF	CCW's and ECB's for Buffer Table Entry
BUFF	Buffer from the File Routine
TAS	Terminal Associated Storage
FSEP	Full Screen Editor Session Table
FCSR	Full Screen Editor Screen Table
FSED	Full Screen Editor Member Table
CPCCB	Cross Partition Communication Control Block
SPL	Spool Parameter List
XPCCB	Control Block for Message Service

Instead of a *table name*, you may enter the following 'commands':

DD hex-addr
Starting at the hexadecimal address *hex-addr*, 256 bytes are displayed in hexadecimal and character formats.

SA hex-addr
Starting at the hexadecim address hex-addr, up to 16 bytes may be **altered**. The area is displayed first. If no alternation is wanted, pressing the ENTER key with no data on the input line will terminate the storage alter mode.

Entering 'DD' or 'SA' with no hex address gives the following three lines:

```
ICCF ADDRESS FROM low-addr TO high-addr
OR SUPERVISOR ADDRESS 0 TO end-of-supv ALLOWED
NO OR INVALID HEXADDR TRY AGAIN
```

The displayed addresses indicate the space that you may address with your 'DD' or 'SA' command.

Instead of having an area displayed, you may request the **scanning**function:

```
SCAN or
SCANF
```

The CICS/ICCF partition will be scanned in a user defined area for a user defined scan argument which may be a character string or a hex string. The maximum length of the scan argument is 16 bytes. Command 'SCAN' will scan the entire area and then display all hits in hex and character format. Command

'SCANF' will display each hit immediately when found. The user can then decide if he wants to continue scanning or not. After entering 'SCAN' or 'SCANF', the following message is printed:

```
K960D  SCAN ADDRESS FROM low-addr TO high-addr
       ENTER SCAN START AND STOP ADDRESS OR EOJ FOR EXIT
```

Now the scan limits must be entered. The following forms are possible:

```
nnnnnn mmmmmmm   scan from address 'nnnnnn' to address 'mmmmmm'
nnnnnn           scan from address 'nnnnnn' to end address
-mmmmmm          scan from start to address 'mmmmmm'
null input       scan from start to end address
```

'nnnnnn' and 'mmmmmm' must be within the scan limits as indicated in the message above. 'mmmmmm' must be higher than 'nnnnnn'.

The following message is printed:

```
K691D   ENTER SCAN STRING OR SCAN STRING WITH FO FOR FIRST OCCURRENCE ONLY
```

You can now enter your scan argument in one of the following formats:

'ABCDEFG'
: Character string which must be enclosed in quotes.

'RRT'RRT'
: Enclosed quotes must be entered twice without blank.

47F0B004
: Hex string, which must be an even number of hex characters.

47XXBXX4
: Her string with unknown halfbytes in the X positions. Any data in the X positions is treated as a match.

50E0BXX8 FO
: The FO sets you in SCANF mode if the scan function was called with 'scan'.

The scan function is the same if the DTSFDUMP is running in a batch partition. The scan function will then be called by UPSI bit 3 (XXX1).

If the scan function is called with 'SCAN' and the FO function is activated by entering a string followed by 'FO', the 'FO' function will be dropped when new scan limits are entered.

To leave SCANF mode, enter an invalid hex character as response to message K960D or K961D. Message K975I will not be issued in SCANF mode.

EOJ

EOJ at any place for any requested input will exit the program.

Regenerate and List (DTSRELST) Utility Program

The DTSRELST utility program takes an existing print area and regenerates print images from that area. The print area may be the actual print buffer area $$PRINT or a library member consisting of a saved copy of a print area.

This program is used by the RELIST macro to reformat print images from the print area for the purpose of reprinting the print lines on the central site printer. In effect, the RELIST macro submits the following job stream for execution under VSE/POWER:

```
   // JOB RELIST
   // UPSI ...
   // EXEC DTSRELST
   name
   @/@
   /*
   /&
   Where name = Either of the following:
```

```
                - $$PRINT
                - The name of a library member that contains
                  a saved print area.
```

The DTSRELST utility may also be used to reformat a saved print area into print lines for display at the terminal. For example, if MEMX is the name of a library member containing a saved print area, the command /LIST MEMX would display that member; however, the format displayed would be the internal print area format. To regenerate the actual print images, you could issue the following commands and statements:

```
 /INPUT
 /LOAD DTSRELST
 /INCLUDE MEMX
 /ENDRU
```

or:

```
$DTSRELST /INCLUDE MEMX
```

The DTSRELST utility uses a UPSI switch (yXXXXXXX) to control the type of analysis that is to be performed on the input data:

- If the UPSI switch is off, the utility interprets the input as normal interactive partition print data. The utility rejoins the multiple 80–byte records to form the original print record of a lenght of up to 156 bytes.
- If the UPSI switch is on, the utility lists the input data as 80–byte records.

Depending on the type of input data (print or not print), the RELIST macro generates a suitable UPSI statement.

Library File Maintenance (DTSUTIL) Utility Program

The DTSUTIL program is the main VSE/ICCF utility for offline file maintenance functions. It is used to backup and restore the VSE/ICCF library file and it can also be used as an archiving facility for the maintenance of offline libraries containing any or all components that have ever been present within the VSE/ICCF online environment.

DTSUTIL also provides several facilities for extracting information from backup files. Not only are directory listings provided when a backup or archive file is created, but also directory listings, member listings or member punch-outs may be obtained from a backup or archive file after it has been created.

The utility is used to back up, archive, and restore the VSE/ICCF library file and to create a new library format. It can also be used to add and delete libraries, library members, user profile and broadcast records, for printing and punching of members or libraries, displaying directory lists, and merging data from previous backup files. Display and clearing of accounting information may also be performed. These functions are described in detail in the following sections.

DTSUTIL normally is run while VSE/ICCF is not active or while the VSE/ICCF library file is disconnected. If, however, VSE/ICCF is active *and* the VSE/ICCF library file is connected, DTSUTIL's access to the VSE/ICCF library file is read-only. That is, no updating or restoring of members, users, libraries or directories will be permitted. If DTSUTIL is run in a VSE/ICCF interactive partition, access to the file is still read-only - with one exception: The ADD/ALTER commands can be used to add or alter user profiles or library characteristics. The BACKUP function cannot be run from an interactive partition.

The DTSUTIL utility gives a return code if // EXEC DTSUTIL,PARM='RC' was specified:

RC=0 for successful execution of DTSUTIL commands
RC=8 if at least one DTSUTIL command has failed.

For the specification of label and extent information for the VSE/ICCF library, refer to the section “Defining the Library File (DTSFILE)” on page 38.

Note: Do not specify SIZE=AUTO in the // EXEC DTSUTIL statement. Preferably, do not specify the SIZE operand at all. If you specify SIZE, its value must be 86K plus an extra space large enough to contain a 26-byte record per member of the library being processed. If a DTSUTIL run involves two or more libraries, the library with the highest number of members determines the size of this extra space.

Command Entry

DTSUTIL commands must begin in column 1 of the card image entered and may extend to column 80. The commands will be read either from SYSLOG or SYSIPT depending on where the EXEC DTSUTIL was encountered. Commands may be abbreviated as shown in the command descriptions. Operands may be separated by spaces, commas or parentheses. Multiple delimiter characters between operands will be treated as a single delimiter. Sequence numbers should not be placed in columns 73 through 80 of DTSUTIL commands because this area will be scanned for operands.

All operands must be wholly contained on a single card. Continuation cards are not allowed. This may necessitate the use of keyword abbreviations. Any abbreviation between the minimum and the full form of the command is accepted.

Any operands in brackets are optional and may be omitted.

The listing which DTSUTIL produces indicates an entered command by the 3-character string '. ' in front of the command.

Restricted Commands Under VSE/ESA Access Control

In a z/VSE system with security active, access to phase DTSUTIL should be limited to an administrator user ID, because several DTSUTIL commands access vital system resources. On the other hand, some DTSUTIL commands (such as PRINT MEMBER) cannot cause any harm.

To protect the more powerful commands of DTSUTIL, z/VSE's predefined security support includes an artificial resource: member IJSYSRS.SYSLIB.DTSUTILA. To be able to use the protected command set, a user must have the proper access right to the DTSUTILA resource defined in the access control table DTSECTAB. For details on the VSE/ESA Access Control function, refer to z/VSE Administration.

The DTSUTIL functions dedicated to resource IJSYSRS.SYSLIB.DTSUTILA are the following:

- FORMAT
- ADD LIB / ALTER LIB
- ADD USER / ALTER USER
- BACKUP
- MERGE
- RESTORE
- DELETE
- PURGE
- SHARE
- DISPLAY .... PASSWRD
- DSERV .... PASSWRD

DTSUTIL UPSI Settings

VSE UPSI switch settings affect the execution of various DTSUTIL functions as follows:

```
// UPSI abcdefgh
```

a = 0 –
Write verify will not be performed on VSE/ICCF library file.

1 –
Write verify will be performed

b = 0 –
Should always be zero.

c = 0 –
Should always be zero.

d = 0 –
DBCS members will not be prepared for printing on an IBM 3200 printer.

e = 0 –
If DTSUTIL is running in an interactive partition, the command input is requested from the terminal. If DTSUTIL is running in a VSE batch partition, the command input is requested from the unit where the EXEC DTSUTIL came from.

1 –
Input for the DTSUTIL command always comes from SYSIPT.

f = 0 –
Print time of day on the output listing as each command is processed.

1 –
Do not print time of day.

g = 0 –
Rewind but do not unload the tape whenever a tape backup or restore file is opened or closed.

1 –
Do not rewind when tape files are opened or closed – the no-rewind option. This option is useful if another file is to be written to a tape following the VSE/ICCF backup or if the VSE/ICCF file is not the first file on the tape.

h = 0 –
Eject the form to the top of the page for DSERV, DISPLAY and PRINT command output.

1 –
Substitute 3 spaces for all page ejects - the save-paper option. This option can shorten some listings produced by DTSUTIL by a few pages. However, the EJECT command always causes a skip to the top of a page. This is useful if only part of the DTSUTIL list output is to be separated.

Formatting the Library File or Changing its Size

FORMAT Command

```
▶▶─ Format ──   ── Users ── ( ── n ── ) ── , ── Libraries ── ( ── n ── ) ─▶◀
```

Format the VSE/ICCF Library File

The FORMAT command is used to create a new VSE/ICCF library file cleared to binary zeros.

The system record is formatted and the specified number of library and user records is allocated. The formatting is done according to the system-generated block size or control interval size. The device type is determined by the assignment of the file DTSFILE. Each track or control interval is built in storage and written to disk at once.

The two operands of the command may be specified in any order.

Users(n)
Determines the maximum number of user profile records allowed on the VSE/ICCF library file. You can specify any number up to 32767. Under the z/VSE Interactive Interface, however, you are limited to 9999 user profile records.

Libraries(n)
Determines the maximum number of library header records allowed on the VSE/ICCF library file. You can specify any number up to 32767. Under the z/VSE Interactive Interface, however, you are limited to 9999 library file records.

Note that specification of LIBRARIES or USERS in the Restore (System) command causes to override the values that were established with the FORMAT command.

The numbers specified here represent the maximum number of users and libraries which may be contained within the library file. Thus, if the number of libraries and/or users will grow in the near future, these values should be made large enough to cope with the anticipated growth. However, it is not necessary to add the actual user profile records or the library records until the need actually arises.

When doing a BACKUP and RESTORE of the VSE/ICCF library file, it would be wise to do a FORMAT between the BACKUP and the RESTORE. This will erase any records beyond the new end of the file. If you leave out the FORMAT, the DTSANALS utility would treat these unerased records as invalid.

With the processing of the FORMAT command, the *number of extents* for the VSE/ICCF library file is determined. This number is derived from and identical to the number of extents specified in the job control EXTENT information for DTSFILE.

Changing the Size of the VSE/ICCF Library

The FORMAT command must be used to change (expand or reduce) the size of the VSE/ICCF library file. The change consists of enlarging or reducing a given extent or of adding or deleting an entire extent.

To change the library size, do the following:

1. Back up the VSE/ICCF library file.
2. Reformat the VSE/ICCF library file by running the DTSUTIL FORMAT command. For this run, the corresponding new EXTENT information must be available, either in the standard label information area or in the job stream.

   When the FORMAT command is used to reformat the VSE/ICCF library file after changing the extent of the file, the FORMAT command *must* be the *first* command following the '// EXEC DTSUTIL'.
3. Restore the VSE/ICCF library file.

To change any of the below listed DTSFILE characteristics, you must first:

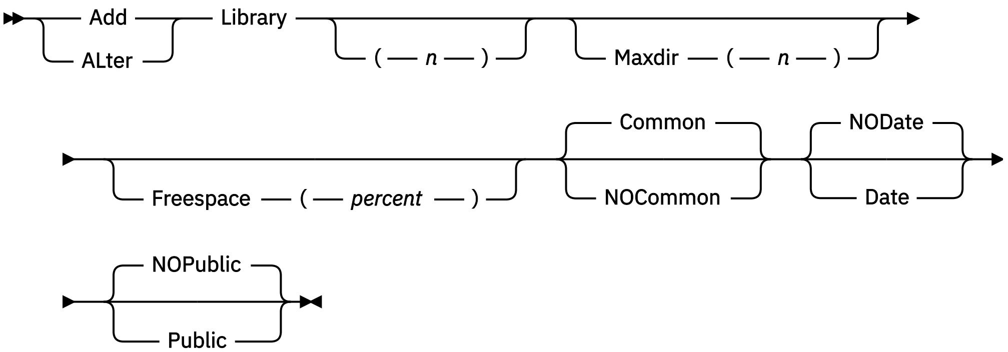
1. Shut down VSE/ICCF.
2. Shut down VSE/POWER if it processes jobs that retrieve SLI members from a VSE/ICCF library.

The affected DTSFILE characteristics are:

The number of extents
The size of extents
The location of extents
The number of records per block
The control interval size (CISIZE)
The number of buffers

Adding or Changing a User Library

ADD/ALTER Command (Library)



Adding a Library

Library

To add a library record, specify LIBRARY as the first operand. This will cause activation of the next available library header record from those allocated by the FORMAT or RESTORE commands. (n), the library number, need not be specified; it is ignored. DTSUTIL returns the number of the new library to the requester when formatting of the record is complete. Once all library records (allocated by the FORMAT or RESTORE commands) are used, it is not possible to add new libraries. However, libraries can be deleted and later made available to new users.

Maxdir(n)

Specifies the number of directory entries allowed for this library. If not specified, it will default to 0, allowing unlimited directory entries. However, it is advisable to limit the number of entries in a directory to a number in the 100 to 200 range. It is more efficient to have several libraries with few members rather than one or two libraries with many members. The maximum value that may be specified for n is 32767.

Note: When a member is added through the ADD MEMBER function, DTSUTIL does not check whether the MAXDIR limit has been reached.

Freespace(percent)

Specifies the percentage of free directory entries. A nonzero percentage allows the number of directory entries in a library to be increased during a RESTORE by imbedding free entries in the directory. The default is zero indicating that imbedded free space is not to be placed in the directory. The highest allowable free space value is 50%, which means that every other directory slot will be available for adding or saving a member.

Imbedded free space may be specified to improve directory lookup performance, especially for directories which are highly active in terms of the saving and purging of new members. New directory entries will be added to the free slots in the same blocks with existing entries, thus eliminating extra disk reads to search for a member. Specifying a large free space for a library which is fairly static may have an adverse effect on performance because it may cause the directory to overflow into a disk block which would not otherwise be required.

Free directory entries are displayed as all asterisks.

For best performance, the percentage of free-space should closely match the expected growth of a library.

Common
NOCommon

If it is desired to prevent directory entries for common data from being placed in this library's directory, specify NOCOMMON. This may be desirable for some users who connect to alternate libraries. They may not want the directory entries repeated for common members in each of their libraries. The default is COMMON. NOCOMMON may *not* be specified for library number 1.

Date
NODate

DATE specifies that a member in this library is to be dated in the directory record with any update to the member. NODATE causes members to be dated only when they are replaced. NODATE is the default.

Public
NOPublic

PUBLIC specifies that this is a public library and may be connected or switched to by any user without regard for the alternate library specification. NOPUBLIC forces the library to private status. NOPUBLIC is the default and ensures that this library must be specified in a user's profile in order to be accessed by that user.

Examples:

```
ADD LIBRARY MAXDIR(90) FREE(20)
A   LIB
A   LIB NOCOM NOPUB DATE
```

Changing a Library To change the specifications for a library after it has been added, specify ALTER LIBRARY (n), where 'n' is the number of the library whose attributes are to be changed. Additional operands may be specified for the library attributes which are to be changed. Any operands not specified will remain unchanged.

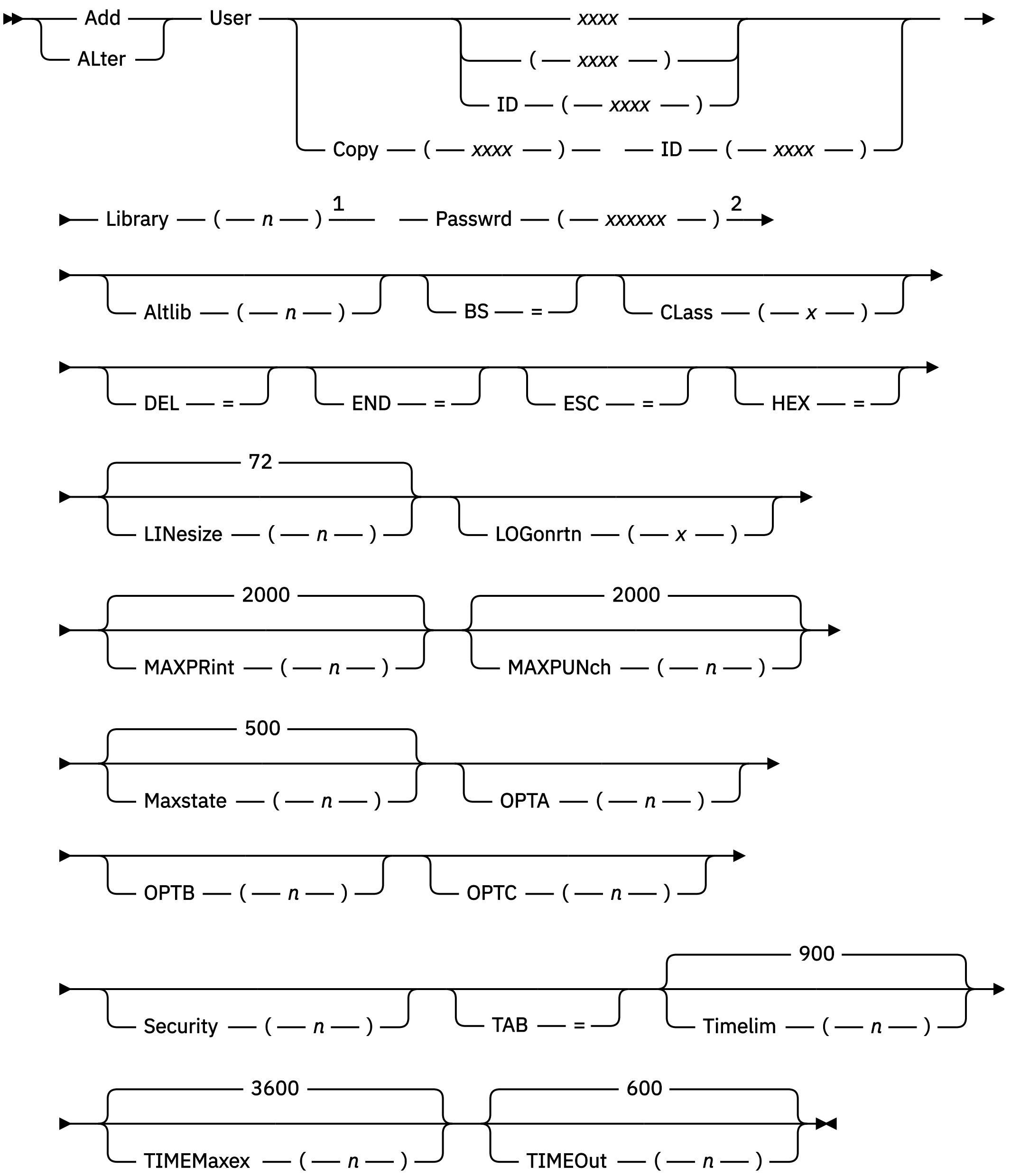
This DTSUTIL command may be issued in an interactive partition, but it may not be issued in a batch partition while VSE/ICCF is up and the VSE/ICCF library file is connected.

Examples:

```
ALTER LIB(4) NOCOMMON PUBLIC
AL     L(3)   FREE(10)
```

Adding or Changing a User Profile

ADD/ALTER Command (User Profile)



Notes:

[1] Library and Password are optional if ID xxxx of the ADD/ALTER command already exists.

[2] Library and Password are optional if ID xxxx of the ADD/ALTER command already exists.

To add or change a user profile record, specify USER as the first operand. Three operands are required when adding a new user: ID, LIBRARY, and PASSWRD. When changing a user profile record, only the ID and the values to be changed need to be specified. If the ID matches a user on the VSE/ICCF library file, the operands supplied replace the specified fields in the user record. If no match occurs, the user is added to the next available user record.

Continuation cards are not supported. It is possible for a new user's profile to exceed one card. In this case, one card will add the profile, and subsequent changes may be made to remove any default values.

Adding a User

Copy

Designates another user whose profile will be used as a model for the new user. If it is specified, it must be the first operand specified after USER. The 'xxxx' operand is the ID of an existing user on the file. This allows a skeleton user profile record to be established with most of the necessary specifications. In this case, the ADD command for each actual user record to be added need contain only COPY, ID, PASSWRD and any other operands that are to be different from the skeleton user record. The ID and PASSWRD must be specified for a new user profile. All other values will be used from the copied user profile unless explicitly specified in the ADD USER command.

ID(xxxx)

Specifies the four-character alphameric identifier which names the user throughout VSE/ICCF. This operand is required if the userid does not immediately follow the keyword USER.

Library(n)

Specifies the library which is to be assigned to the user. (The library can be owned by the user, or it can be shared with other users.) This operand is required for a new user profile if COPY is not specified. 'n' must be a decimal number indicating one of the libraries which have previously been added to the file. The same library number must not be specified as a user's primary and alternate library. Therefore, a user's primary library must not be altered to a number which is contained in the user's profile as an alternate library. Likewise, a primary library cannot be made an alternate library.

By an intermediate alter step, however, this can be achieved, as shown in the following example:

```
ADD U ID(ANNA) P(J271) L(3) A(4 6 9)    (original definition)
AL  U ID(ANNA) P(J271) L(3) A(4 9)      (intermediate alter)
AL  U ID(ANNA) P(J271) L(6) A(4 3 9)    (changed profile)
```

Passwrd(xxxxxx)

Specifies the three to six character alphanumeric password which the user must enter for identification during logon. The specified password is padded with blanks on the right if it has less than six characters. This operand is required for a new user profile.

Altlib(n)

Specifies a list of one or more libraries to be associated with this user. It is invalid to specify public libraries because these are available to all users. If this operand is not specified, the user has access only to the library specified in the LIBRARY operand and to public libraries. Up to eight alternate libraries may be specified. These libraries may be owned by the user or shared with others in any combination. The terminal user must issue a /SWITCH or /CONNECT command in order to access one of these libraries. See also the description of the LIBRARY operand above.

BS=

Is described in the next to the following paragraph.

Class(x)

Specifies one alphabetic character which is used to determine in which interactive partition to execute if no class is set by the user. The default is null which is equivalent to A. Whenever an interactive partition execution is requested, VSE/ICCF attempts to find an available interactive partition such that the user's class matches one of the classes in the interactive partition. Via this operand, the default class for the user may be assigned.

BS= DEL= END= ESC= HEX= TAB=

Specify a single character which is used to set one or more of the user's control characters. The control characters specify the user default values for the logical backspace character, the line delete character, the logical line end character, the escape character, the hexadecimal entry character, and the logical tab character. For more information on these functions, see the SET Command in the VSE/ICCF User's Guide. For the duration of a terminal session the control characters can be altered with the /SET command. The default characters must not be alphabetic or numeric and may not be any of the delimiter characters (parentheses, comma, or space) or any of the special characters

```
$  @  *  /  =  '(quote)
```

Every control character has a default of null if not specified. A character may be reset by specifying the keyword OFF. It is not permitted to assign the same character to more than one function. The evaluation for duplication is made after all operands are processed.

LINesize(n)
Specifies the width of the data display for system command functions such as /LIST and /DISPLAY. The default is 72. Specify a value from 1 to 80. This value, if greater than 72, will also be used to indicate the default for the PRINT and VERIFY commands when the editor is entered.

LOGonrtn(x)
Specifies the name (up to 8 characters) of a macro or procedure to be executed when the user logs on. Such a macro or procedure may be used to set up user defaults, display mail, display a greeting on the terminal or some other logon-time function. If a macro of the form DTSxx&& is specified, a form of device independence may be achieved. When processing the DTSxx&& macro, VSE/ICCF replaces the characters && with one of the following:

70
For an IBM 3270

41
For an IBM 2741

40
For an IBM 2740

00
For any other device type

Thus, if a user may log on to more than one terminal type, a logon procedure may be tailored to the type of terminal. The 'xx' in the name may be any two characters. Similarly, if a procedure is specified and it is of the form DTSxxx&&,; the && will be replaced as above. To remove a previously entered logon operand, specify LOGON(RESET) on the ALTER command.

MAXPRint(n)
Specifies the upper limit of the print area ($$PRINT) in terms of lines printed. The /SET BUFFER command will not allow this value to be exceeded. The default is 2000 lines. The maximum specification is 9999.

MAXPUNch(n)
Specifies the upper limit of the punch area ($$PUNCH) in terms of card images punched. During execution punch records will be allocated up to this value. The default is 2000 card images. The maximum specification is 32767.

Maxstate(n)
Specifies the maximum number of statements that the user's input area may contain. If during restore of a library member this value is exceeded, a message will be printed and the member may optionally be truncated at the limit. If this operand is not specified, it will default to 500 for a new user and will not be changed for an existing user. The maximum value is 9999.

OPTA(n) OPTB(n) OPTC(n)
Are user option flag bytes. These operands cannot be abbreviated. The value for each of these operands is a character string one to eight digits long. The string is a character representation of the binary value of the flag byte. Any digits not specified will be padded on the right with zeros. If these operands are omitted, the fields will be set to binary zeros for a new user and they will not be changed for an old user.

Following are the descriptions of the switches. A '1' indicates that the option is on, and a '0' indicates that the option is off. The switches below are described from left to right. *OPTA Option Switches:*

Bit
Explanation

0 = 0 –
Inclusion prompting is not the default for the user.

1 –
Inclusion prompting is the default for the user. Unless specified otherwise, the terminal line number prompt will become part of the data line.

1 = 0 –
Normal.

1 –
This user may bypass access control check on library member password. That is, he may access a member without specifying the password.

2 = 0 –
Normal.

1 –
This user may bypass access control check on write-type access to a member. That is, he may update a private member even though his userid does not match the identifier of the user who entered the data.

3 = 0 –
Normal

1 –
This user may bypass access control check on read-type access to a member. That is, he may read a private member even though his userid does not match the identifier of the user who entered the data (alternate security only).

4 = 0 –
Normal

1 –
If this switch is set, this user may not run jobs in the background, that is, in interactive partitions. This means, he may not use the /EXEC, /ENDRUN or /RUN commands. He may, however, use the SUBMIT procedure for submitting to batch execution via VSE/POWER.

5 = 0 –
When this switch is off, the user's library members will be saved as public unless specified as private in the /SAVE command.

1 –
The user's library members will be saved as private, except if specified otherwise in the /SAVE command.

6 = 0 –
Prompting with line numbers on input is not the default for the user.

1 –
Prompting with line numbers on input is the default for the user.

7 = 0 –
When a user submits a job to batch processing by VSE/POWER, any VSE/POWER JECL operand that he submits with the job is monitored against the installation model JECL (see the description of the DTSSUBMT program earlier in this chapter, in particular the IBM-supplied SUBMIT procedure which is listed in Figure 16 on page 87). Defaults are added, and restricted values are overridden, if necessary. Operands that are not allowed to be specified (parameter=*** in the model JECL) are rejected.

1 –
This user's JECL is passed on to VSE/POWER without being subject to installation-established overrides. The user is allowed to specify also 'not-to-be-specified' operands (parameter=*** in the model JECL).

OPTB Option Switches:

Bit
Explanation

0 = 0 –
Normal.

1 –
The user will not undergo a forced logoff, even if his terminal timeout value expires. Normally, if no activity is received from a terminal within the time specified in the TIMEOUT operand, /LOGOFF is forced.

1 = 0 –
Normal.

1 –
The user will be able to access VSE volumes and files regardless of what is specified for the volume or file in the system file table.

2 = 0 –
Normal.

1 –
This user is the VSE/ICCF administrator. When this bit is on, the user will be able to perform certain types of functions not available to ordinary users. Among these functions are: setting off editor flagging, overlaying editor flags, setting off update-in-progress flags from the terminal, scratching dynamically allocated space by location, requesting DTSAUDIT reports at the terminal and issuing VSE/POWER commands from the terminal via the /CTLP command. Any user for whom this bit is on may control jobs in any one of the VSE/POWER queues.

3 = 0 –
Normal.

1 –
The user will be able to switch (/SWITCH or /CONNECT) to another library regardless of whether he is authorized to use the library. This switch should be on for the VSE/ICCF administrator. It should probably be off for other users, but this decision is up to the installation itself.

4 = 0 –
Normal.

1 –
The user will be able to access programs restricted to authorized users only. If a program is flagged as authorized in the System Program Table, in order to access the program, the user must have this switch on.

5 = 0 –
The user may not issue the /PASSWRD command.

1 –
The user may issue the /PASSWRD command to change his logon password.

6 = 0 –
The user may not update common data.

1 –
The user may update common data. This right to update should be reserved for the user ID associated with the VSE/ICCF administrator.

7 = 0 –
VSE/ICCF's SUBMIT procedure will modify a VSE/POWER JECL statement according to the VSE/ICCF user's setting, even if columns 73 through 76 contain $SLI.

1 –
The entire 80 columns of a VSE/POWER JECL statement will be submitted unmodified by the VSE/ICCF SUBMIT procedure if the contents of columns 73 through 76 is $SLI.

OPTC Option Switches:

Bit
Explanation

0 = 0 –
Messages will not be displayed automatically.

1 –
Messages will be displayed automatically. If remote terminals are used, automatic message display should not be requested. The same is true if a short-on-storage condition can easily occur on your system.

1 = 0 –
The VSE/POWER JECL statements * $$ LST and * $$ PUN are automatically inserted into a submitted job stream.

1 –
No VSE/POWER JECL statements are inserted if the * $$ JOB statement contains the XDEST operand (this allows a job stream to be submitted unchanged to a non-VSE/POWER node).

2 = 0 –
Normal. Only the VSE/ICCF administrator may enter the commands /CP and /CTLP.

1 –
The commands /CP and /CTLP may be issued also by non-administrator users.

3 - 4 –
Should always be 0.

5 = 0 –
The user may allocate permanent file space within a dynamic space area via the DISP=KEEP operand of the /FILE statement.

1 –
The user may not allocate permanent file space but still may allocate temporary space.

6 = 0 –
Normal. The user may specify any phase name on a /LOAD statement as long as this name is not a restricted entry in the system program table and the phase is stored in a VSE sublibrary.

1 –
The user will be able to access only those programs which are specified in the system program table. The VSE access control function is not affected by this bit.

7 = 0 –
Normal.

1 –
The user may not issue the /GROUP command to create or remove generation member groups in or from his library.

Security(n)
Specifies the user's access control class as a number from 1 to 32. The default 0 indicates that the user has no access control class. If the user attempts to access a secured volume, a secured file or a secured program, the user's access control class (1 to 32) must match one of the classes specified in the SECURITY operand for the program, the file or the volume. See "System Program Table" on page 47, "System File Table" on page 51, and "The VSE Access Control Table (DTSECTAB)" on page 54.

Because continuation statements are not permitted for utilities, in certain situations the ALTER command must be used to alter or add access control classes. The following example shows how a user is added:

```
Add   U ID(ICCF) L(1) S(1 2 3 4 8 9 10 11 12)
ALter U ID(ICCF) S(25 26 28 30)
```

To delete a class, specify the ADD command with the same access control class as that currently in effect for the user, but leave out those which are to be deleted. For example, to delete class 12 and add class 32 in the above example, specify:

```
Add   U ID(ICCF) S(1 2 3 4 8 9 10 11 25 26 28 30 32)
DIS   U ICCF
```

The DISPLAY command is used in this example to show the results of the ADD command.

TAB=
Is described together with BS= DEL= END= ESC= HEX=, above.

Timelim(n)
Specifies the user's default time limit in 'execution units' which is set each time a job is initiated. An execution unit is approximately equal to one second of wall clock time while the job is actually executing. A user may vary this value via the /SETIME command or the TIME= operand of the / OPTION statement. If the default time limit value or an explicitly specified time limit is exceeded during execution, the job will be canceled. If not specified, the default is 900. The maximum value which may be set is 32,767.

TIMEMaxex(n)
Specifies the maximum amount of time in seconds an execution may tie up an interactive partition before it is automatically canceled. This time includes time spent waiting for conversational reads and time for reviewing printed output and thus differs from the TIMELIM value which only specifies time in actual execution. The default value is 3,600 units. The maximum value which may be specified is 65,535. The value specified here may be altered by the user via the /SETIME command or the / OPTION job entry statement.

TIMEOut(n)
Specifies the amount of time in seconds which may elapse between terminal input operations. When this time has passed, the user is automatically logged off, except when bit 0 of the user option flag byte OPTB is on. The default is 600 seconds; the maximum value that you can specify for n is 3,600. Do not define a timeout for any of the VSE/ICCF users in the sign-on table of CICS. A log-off forced by a CICS controlled timeout can endanger the integrity of the affected user's data.

Examples:

```
ADD USER ID(STUA) PASS(STUAPS) LIB(4) LOGON(DTSLP&&)
ADD USER COPY(STUA) ID(STUB) PASS(STUBPS)
AL U STUC P(STUCPS) SEC(6) CL(A) LIN(80) ALT(2,3,6,10) LIB(1)
AL U STUD P(STUDPS) L(13) DEL=# END=> TAB=; MAXPR(2500)
```

Changing a User Profile

To change a user profile, specify the command as ALTER USER. The ID operand is required to specify which user profile is to be updated. The keyword ID may be omitted if the userid immediately follows the keyword USER. Refer to "ADD/ALTER Command (User Profile)" on page 129 for a detailed description of the operands which can be altered. The changed user profile becomes effective when the user does a new logon with the updated profile.

The ALTER USER command of DTSUTIL may be issued in an interactive partition, but it may not be issued in a batch partition while VSE/ICCF is up and the VSE/ICCF library file is connected.

Examples:

```
ALTER USER ID(AAAA) OPTA(01110001) SECURITY(8)
AL U STUA TAB=; END=! CLASS(Z) TIMEOUT(1800)
AL U STUB PAS(NEWPAS) ALT(2 4 6 8 10)
AL U STUC LOGONRTN(RESET) HEX=OFF TAB=OFF SEC(7)
```

Special z/VSE Considerations for Adding a User

z/VSE has a special structure of user profiles. For one, there are three *user types*:

- **1** Administrator. This type provides VSE/ICCF administrative authority.
- **2** Programmer. This type provides VSE/ICCF access, but not administrative authority.
- **3** General. This type does not provide VSE/ICCF access. It is used for application end users.

The discussion of ADD USER parameters below highlights the values you can use. It also recommends which defaults you should keep. You should carefully consider the recommended values and any changes that you make. This is to ensure that the Interactive Interface operates correctly.

- MAXSTATE - The value must be greater than 100.

- OPTA, OPTB, OPTC

  The default option byte settings (OPTA, OPTB, and OPTC) are based 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. The bits that you can change are described below and are shown by an asterisk (*).

  - 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 (Programmer and Operator)

    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)

  As a general rule, you should only change the bits that are shown with the *. If you change any other bits, the Interactive Interface may not work correctly for that user except for the following case:

  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.

- Defaults.

  Do not change the defaults for:

  - BS
  - DEL
  - END
  - ESC
  - HEX
  - LOGONRTN
  - TAB
  - TIMEOUT

Adding Broadcast Records or Library Members

ADD Command for Broadcast Records

```
►► Add ──    ── Broadcast ─►◄
```

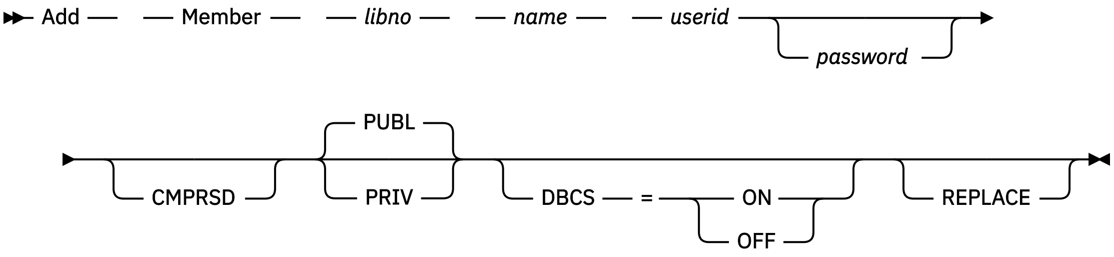
Broadcast
To add or change broadcast records, specify BROADCAST as the only operand. The BROADCAST operand may be abbreviated to one character if desired.

Have the command followed by up to eight new broadcast records. These new records replace any previous ones. Each submitted broadcast record must have an * in the first column. If a record does not, VSE/ICCF assumes it to be a command to DTSUTIL. Broadcast records are padded on the right with blanks if they are shorter than 80 characters.

A broadcast record must not contain double-byte characters.

After having entered the broadcast records, enter any DTSUTIL command, except a comment (*) command. This signals the end of the broadcast records.

ADD Command for Library Members



The ADD MEMBER can be issued only when VSE/ICCF is not active or when the DTSFILE is disconnected. Specify the operands you use in the order as shown above.

Member
Indicates that a new member is to be added to the VSE/ICCF library file.

libno
Specifies the library which is to receive the new member.

name
Specifies the name of the new member. The name must be a string of 1 to 8 characters, and its first character must be an UPPERCASE alphabetic character. The other characters may be alphanumeric, but DTSUTIL will translate an alphabetic lowercase character into uppercase. The member will not be added if a member of the same name already exists on that library.

userid
Is the four-character userid to be associated with the member.

password
Is an optional operand and, if present, specifies a This password may be one to four characters, the first of which must be alphabetic; it must not be PRIV or PUBL.

CMPRSD
Indicates that a compressed-format member was punched from the VSE/ICCF library file by DTSUTIL and that this member is now being re-added to the file or added to a different file. Normally, this keyword is specified only in a REPRO ADD MEMBER command that precedes a PUNCH request for a compressed member. This keyword is placed in the ADD MEMBER command if it is required and if the command was created by the DTSUTIL command PUNCH with the PUNCTL option.

PRIV
PUBL
Indicates that the member is to be added as private or public data of the specified user (userid). Omission of this keyword indicates that the member is to be added as public data.

DBCS=
Allows you to add the member with the DBCS attribute being set for the member. If you specify DBCS=OFF or omit the operand, the attribute is not set.

REPLACE
Performs a purge member and a subsequent add member. No return code is being issued if the member does not exist initially. This operand has been introduced to ship new members via IBM Service.

The records for the member must immediately follow the command. The last record in the member must be followed by END OF MEMBER in the first 13 columns which will signal the end of the member. The string END OF MEMBER must therefore not appear anywhere else in the member. The new member must not contain statements beginning with '/*' or '/&' because they signal the end of processing by DTSUTIL. Use '../*' and '../&' instead of '/*' and '/&'. When DTSUTIL finds '../*' or '../&' statements on input, it converts them before it places them into the VSE/ICCF library file. Similarly, when you run DTSUTIL under VSE/POWER, the member to be added must not contain any VSE/POWER JECL statements beginning with '* $$ '. These statements are accepted only if they begin with '..$$ '; DTSUTIL converts them into statements beginning with '* $$ ' before it adds them to the VSE/ICCF library file.

Note: When a member is added through the ADD MEMBER function, DTSUTIL does not check whether the MAXDIR limit has been reached. The MAXDIR operand is described in section "ADD/ALTER Command (Library)" on page 126.

Figure 18 on page 138 shows an example job for the use of the ADD command of DTSUTIL. (DLBL/EXTENT information for the DTSFILE is not shown because it is assumed to be stored in the standard label information area.)

```
// JOB    BUILD FILE
// EXEC PROC=DTRICCF            ASSIGN FOR DTSFILE
// EXEC   DTSUTIL
FORMAT    LIB(40) USERS(30)
ADD  LIB FREE(25)               ADD LIBRARY NO. 1
ADD  LIB MAX(200)               ADD LIBRARY NO. 2
ADD  LIB MAX(200) FREE(25)      ADD LIBRARY NO. 3
ADD  USER ID(AAAA) LIB(1) PAS(D$ADMN) OPTA(01110111)
AL   USER ID(AAAA) OPTB(11111110)
ADD  USER ID(UCDA) LIB(3) PASS(AWLNOV) MAXST(800) SEC(5 8 16 32)
ADD  BROADCAST
* AUGUST 22, 1992.  SYSTEM WILL SHUTDOWN AT 6 PM.
ADD  MEMBER 2 A$MAIL ADMN
    VSE/ICCF GENERAL MAIL A NEW
    UTILITY HAS BEEN ADDED TO THE ETC., ETC.
UCDA SPECIFIC MAIL FOR USER UCDA
UCDA BACKUP AND RESTORE THE LIBRARY FILE
END OF MEMBER
/*
/&
```

Figure 18. Sample Job – Use of the ADD Command of DTSUTIL

Backup and Restore the VSE/ICCF Library File

When reading about the commands for backup and restore, be aware that z/VSE has dialogs for these tasks. How to work with these dialogs is described in z/VSE Operation.

BACKUP Command

▸▸— Backup —▸◂

The BACKUP command causes all records of the VSE/ICCF library file to be written to a backup file on disk or tape (file name DTSBKUP).

The label information and ASSGN for DTSBKUP determine the output file. If a tape backup file is used, it must be assigned to SYS005.

The records are written to the backup file in the following order:

System record
User records
All library header records
Broadcast records
The first library header record
All common members (if any)

All non-common members of the first library
The next library header record
Its non-common members.

This last set of record types is repeated for each active library header record in the VSE/ICCF file. After the last member is written, the user records are written again to the backup file with updated accounting information on space utilization. This same information is also added to the user records on the VSE/ICCF file.

In addition, a directory listing with the number of records in each member is produced after each library is written to the backup file.

The backup function can be run when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected.

It is possible to create a backup file on tape and not rewind at close. For more information refer to "DTSUTIL UPSI Settings" on page 124.

Note:

1. Use the file-id operand in the // DLBL statement when backing up to disk. This file-id is required if you later want to perform a MERGE, RESTORE, or INPUT operation from this backup file.
2. Use the date operand in the // TLBL statement to make sure that alternate tapes are not overwritten during backup.

MERGE Command

```
▶▶─ Merge ─▶◀
```

The MERGE command provides an archive facility for the VSE/ICCF file. Members which have been purged from the VSE/ICCF file can be retained on the backup files through the use of the MERGE command. The MERGE function combines a normal VSE/ICCF file backup with a comparison between the input backup file (DTSMERG) and the VSE/ICCF library file. The result of the comparison is a file in standard backup format which contains all of the members on the VSE/ICCF library file and in addition all members from the input file which were not also present on the VSE/ICCF library file. Comparison is made by library number and member name. The MERGE function can be run when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected.

The MERGE function allows inactive members to be retained in the backup files while not requiring valuable online space. A member can be recalled from the archive file by using the RESTORE MEMBER command. If a full restore of an archive tape is performed, only the active online members will be restored (unless RESTORE ALL is specified). Thus, it is possible to follow a MERGE operation with a RESTORE operation to accomplish the reorganization function recommended in the VSE/ICCF start up procedure. The label information and ASSGN of DTSMERG and DTSBKUP determine the input file and the output file, respectively. Both files must be located on devices of the same device type, for example both on tape, or both on IBM 3380. If TAPE is specified or assumed, the input file must be on SYS004 and the output file on SYS005. Both files must have standard labels. Refer to "DTSUTIL UPSI Settings" on page 124 for a discussion of the no-rewind option.

The directory listing produced by the MERGE is similar to the directory listing produced by DSERV or BACKUP, except for two areas:

1. The number of records in a member is not available for members from the input backup file.
2. The field AF in the first column of the listing contains a flag whose meaning is explained below:

   N

   Indicates that the member was not found on the VSE/ICCF file but was on the input backup file in the active status indicating that the member was on the VSE/ICCF library file at the time the backup archive file was created. However, between the last backup or archive function and the present one, the member was purged from the VSE/ICCF library. The member will be transferred from the input backup (archive) file to the output archive file and will be changed to inactive status.

*****

Indicates that the member was found on the VSE/ICCF library file and on the backup (archive) file; however, the member on the backup file was found to have inactive status. This may mean that either the member was restored to the file (via RESTORE MEMBER or RESTORE LIBRARY) since the last backup or that a new member with the same name was added to the file (in the same library). In the second situation, the original member will be lost and the new member with the same name will become the named member on the archive file. This can be a problem. Therefore, whenever the * flag appears, be certain that the condition is valid before you discard the input backup file.

C

Indicates that the member was not found on the VSE/ICCF library file and was in the inactive status on the input backup file. The member is copied from the input to the output backup file without change. This is the normal flag for archived members.

blank

Indicates that the member was on the VSE/ICCF library file and was either not on the input backup file or was on the input backup file in an active status. The member is taken from the VSE/ICCF library file and written to the backup file, and the copy of the member on the input backup file is bypassed. This is the normal flag for VSE/ICCF active online members.

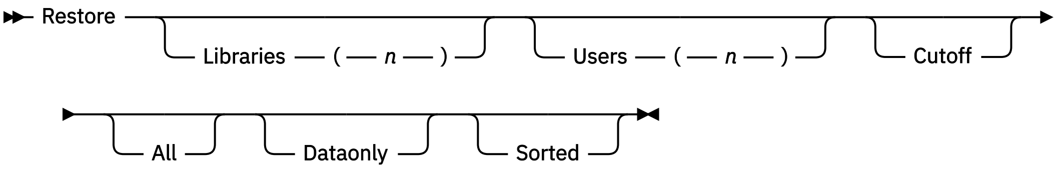
SKIPRMSG Command

```
►► SKIPRMSG ►◄
```

The SKIPRMSG causes all following RESTORE functions to proceed without issuing message

K238D RESPOND 'GO' TO RESTORE / 'NOGO' TO IGNORE

RESTORE (System) Command



The RESTORE function can be run when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected.

This format of the RESTORE command may be used to read a backup file and restore the records from the backup file into a pre-formatted VSE/ICCF library file. If the VSE/ICCF library file into which the backup file will be restored does not already exist, the FORMAT command must precede the RESTORE. Or, if the file size, location or number of extents must be changed, it is also necessary to use the FORMAT command prior to the RESTORE.

When the VSE/ICCF library file is being restored, it is possible to alter the maximum number of user profile and/or library allocations.

The label information and ASSGN of DTSRSTR determine the backup file. If tape is used, then it must be assigned to SYS004. The number of library records and user records may be optionally altered in the same manner as on the FORMAT command (see "FORMAT Command" on page 125). If the LIBRARIES and USERS operands are not present, the values on the backup file will be used. Do not specify the DATAONLY operand (see below) together with the LIBRARIES or USERS operand.

The utility ensures that the new specifications do not eliminate any users or libraries represented on the backup file. That is, the minimum specification for users and libraries is the number of active users and libraries which were written to the backup file. Any attempt to reduce these numbers results in an error message and ends the restore run. If the RESTORE (System) command is submitted without the DATAONLY operand, the complete VSE/ICCF library file is restored.

A message will be written to the console indicating the date and time of the backup. The operator should verify that the correct backup file is being restored. If the message indicates the correct file, the operator should reply 'GO' to continue with the restore. A reply of 'NOGO' terminates the restore.

The meaning of the command operands is as follows:

Libraries(n)
Determines the number of library header records allowed on the VSE/ICCF library file. This number cannot be smaller than the number of libraries contained in the backup file. The maximum number you can specify for n is 32767.

Cutoff
Requests the immediate truncation of any member that exceeds the user maximum record count as indicated in the user profile record. The remainder of the member is bypassed and the restore continues. A warning message is always written when this limit is exceeded even if the option is not specified.

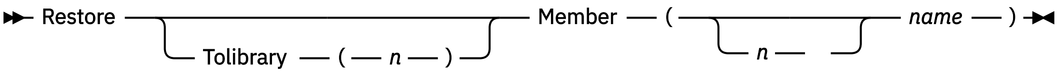
Users(n)
Determines the number of user profile records allowed on the VSE/ICCF library file. This number cannot be smaller than the number of users contained in the backup file. The maximum number you can specify for n is 32767. But keep in mind that the use of the z/VSE Interactive Interface imposes a limit of 9999 users.

All
Specifies that all the data on the VSE/ICCF library file backup be restored. *This includes all archived data* (see the "MERGE Command" on page 139 for an explanation of archived data). Note that archived libraries that have a higher library number than the library with the highest number on the restore file, will not be transferred to the restore file. If ALL is omitted, then only those members copied from the VSE/ICCF file onto the backup (that is, the active online members) will be restored. This is the normal use of the restore function.

Dataonly
Specifies that the system record, the user records and the library header records will *not* be restored. Only library member and broadcast records will be restored from the backup file. The primary use of this operand is for restoring a backup file from the original ETSS II. User and library records must be built first with the DTSUTIL commands ADD USER and ADD LIBRARY if they do not already exist within the newly restored VSE/ICCF library file. The DATAONLY option might also be useful when restoring a backup file whose user profile or library header records have been altered since the backup file was created.

Sorted
Specifies that the non-common members in each library are to be restored to the directory in alphabetical sequence.

RESTORE (Member) Command



RESTORE can be run when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected.

This form of the RESTORE is used to copy one member from the backup file to the VSE/ICCF library file.

Tolibrary(n)
Indicates the number of the library which is to receive the new member. If the operand is omitted, the library number from the backup file will be used.

If the library in which the member appears on the backup file is no longer present within the online VSE/ICCF library file, then the TOLIBRARY operand must be specified and must be specified prior to the MEMBER operand.

Member(name)
Member(n name)

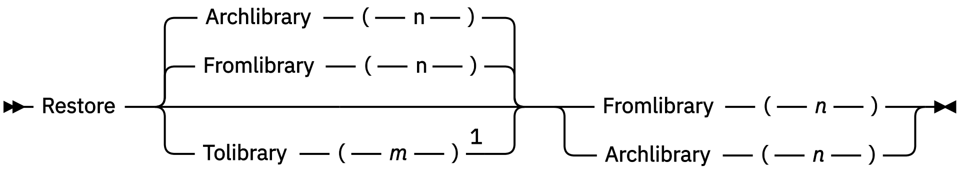
For n, specify the number of the library in the backup file which is to be searched for the named member. If you do not specify a library number, the backup file is searched from the beginning for the first match on 'name'. It is also the library number in the VSE/ICCF library file which receives the member unless TOLIBRARY is specified.

If the member already exists in the specified library, it will be deleted prior to the restore. If the member being replaced was marked as common member, the restore will not be done. The common member must first be purged. Then, the restore can be done and the new copy SHAREd for all users.

If it is planned to selectively restore more than one member, the RESTORE commands should be in ascending order by library number. Also, if possible, the member names should occur in the same sequence as the members were backed up. The restore does not require this sequencing, but will take advantage of it by eliminating extra search passes of the backup file.

If this format is used, *do not* specify a no-rewind (refer to “DTSUTIL UPSI Settings” on page 124).

RESTORE (Library) Command



Notes:

[1] Must be specified if Fromlibrary(n) or Archlibrary(n) does not exist on the DTSFILE.

RESTORE can be run when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected.

This form of the RESTORE allows an entire library to be restored. All non-common members already in the receiving library are deleted. Then the members from the backup are restored.

Fromlibrary(n)

Specifies the number of the library on the backup file which is to be restored. The FROMLIBRARY specification indicates that only non-archive members (that is, active online members) are to be restored. These are the members resident on the online VSE/ICCF library file when the backup was taken. The library number is required and will also be the receiving library if TOLIBRARY is omitted.

Archlibrary(n)

Also specifies, just like the FROMLIBRARY operand, the number of the library on the backup file which is to be restored. However, *all* data from this library will be restored including archive data.

Tolibrary(n)

Specifies the number of the receiving library. In cases where it is desired not to disturb existing libraries, this would be the number of an unused (but previously established) library record. If the TOLIBRARY operand is omitted, the library number of the FROMLIBRARY or ARCHLIBRARY is used as the receiving library.

If the FROMLIBRARY (or ARCHLIBRARY) is valid on the backup file but no longer exists within the online VSE/ICCF library file, the TOLIBRARY operand must be specified and must be specified in front of the FROMLIBRARY operand.

Note that names are not checked for duplications. At the completion of the restore, it may be possible to have duplicates between common and noncommon data. A rename will solve this problem in most cases.

If this format of the RESTORE is used, *do not* specify a no-rewind (for more information, refer to “DTSUTIL UPSI Settings” on page 124).

Note:

1. Selective restore of one or more libraries will normally cause severe performance degradation because directory and member records will be intermixed after restore. Also, the DISPLAY USER command of DTSUTIL will give incorrect file statistics after a selective restore. It is highly recommended to perform a backup and restore of the *complete* VSE/ICCF library file after a selective restore.
2. The FREESPACE factor (which is described in "ADD/ALTER Command (Library)" on page 126) is ignored when processing the RESTORE Library command.

Examples:

RESTORE
The previous LIB and USER allocations are maintained.

RES LIB(6) CUT
The previous LIB allocation is altered. All members that exceed the maximum record number as indicated in the user profile are truncated.

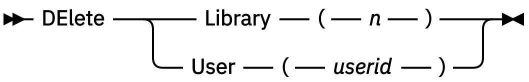
RESTORE MEM(13 MYMEMBER)
A backup file on DTSRSTR is searched for library 13, member name MYMEMBER. If found, the member is restored into library 13 of the VSE/ICCF library file.

RESTORE FROM (6) TO (10)
A backup file is searched for library 6; all non-common members are deleted from library 10; non-archive members from library 6 on the backup file are restored to library 10.

RESTORE MEM(WHERSIT) TO (8)
The backup file on DTSRSTR is searched for WHERSIT. When the member is found, library 8 on the VSE/ICCF library file is searched for WHERSIT. If the member exists in that library, this member is deleted. The member is then restored from the backup file.

Library-File Maintenance

DELETE Command



The DELETE command is used to clear a specified library header or user record.

The DELETE command can be used when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected.

Library(n)
Requests the pointer to the first directory record in library n (except library 1) to be deleted if no user is being assigned to that library. If a user is assigned, an error message is printed and the pointer remains unchanged. Library 1 may not be deleted.

If the DELETE command is accepted, the library header record is made inactive.

The directory and member records that were part of the deleted library are not attached (chained) to anything after the delete run. These records become part of either the high file area after the next BACKUP/RESTORE reorganization, or of the free chain after the next RECOVER or REORG run of DTSANALS.

Since no pointers are moved and no member records are read, a delete run is significantly faster than a purge run. This is the main reason for using DELETE LIB (n) in place of PURGE LIB (n) ALL.

Before the DELETE LIBRARY command takes effect, all users of that library must be deleted using the DELETE USER option. Alternatively, the user profiles may be changed to specify another library as the primary or alternate.

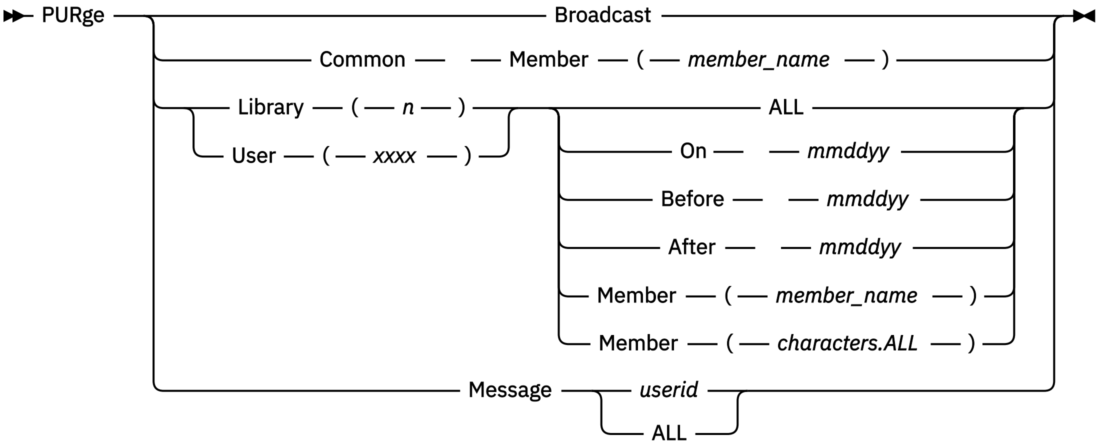
User(userid)

Invalidates the specified user's id in the user profile record. All members in the VSE/ICCF library file associated with that user are unchanged. Deleting a user only prevents that userid from being used to log on. At a later time, the userid may be re-added to the file; the connection between the userid and the user's original data will be reestablished. If you wish to not only delete a user but also delete all of his data, the PURGE USER ALL command should precede the DELETE USER.

Example:

```
DEL         L(17)
DELETE      USER(ABCD)
```

PURGE Command



The PURGE command is used to selectively delete members from their libraries and place the deleted records in the free chain. The command can be used when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected.

Broadcast

Requests purging of the broadcast records. Specify BROADCAST as the only operand. The broadcast records will be placed in the free chain, and the pointer to the first broadcast record will be cleared.

Common Member (member_name)

Requests purging of the common member 'member_name'. Specify COMMON as the first operand. The second operand must be MEMBER. The third operand is the name of the common member to be purged.

This format is the only way to remove a common member. All references to the member will be removed from every directory and the records of the member will be placed in the free chain.

Library(n)

Requests purging of members from a specific library. Specify LIBRARY as the first operand. The second operand must be the library number (n). Common members are not deleted when selection is by library. The third operand determines the selection method for the members:

ALL

Specifies that all members in the library are to be deleted; the library itself is not deleted. Any common members within the library will remain in the library. New members may be added following the PURGE because the library is still active.

ON or BEFORE or AFTER mmddyy

Specifies that selection is by the date of the last modification or of the save or replace of the member. The date in the operand is compared with the modification date in the directory record. Members whose modification date falls within the specified time period are purged. This command accepts both 2- and 4-digit-year input. VSE/ICCF uses the YEAR224 macro (-74+20) to convert 2-digit entries to 4-digit values relative to the current year.

Member (member_name)
Member (characters.ALL)

Specifies that the selection is to be by member name, either by the full member name or by a generic name (1 to 7 characters.ALL). The library is searched for a member matching the specific name or members whose first characters match the characters in the generic name. When a match is found, the member(s) is (are) deleted.

User(xxxx)

Requests purging of members that belong to a specific user. Specify USER as the first operand. The second operand must be the user identification code. Common members are not deleted when selection is by user. The third operand determines the selection method for the members:

ALL

Specifies that every member in the VSE/ICCF library file whose user identification matches the second operand will be deleted.

ON or BEFORE or AFTER mmddyy

Specifies that selection is by the date of the last modification or of a save or replace for all members whose user identification matches the second operand, 'xxxx'. Members with the matching identification whose modification date falls within the specified time period are purged. This command accepts both 2- and 4-digit-year input. VSE/ICCF uses the YEAR224 macro (-74+20) to convert 2-digit entries to 4-digit values relative to the current year.

Member (member_name)
Member (characters.ALL)

Specifies that the selection is to be by member name for the specified user. The name is either a full member name or a generic name (1 to 7 characters.ALL). All libraries are searched for a member matching the specific name or members whose first characters match the characters in the generic name. When a match on name and userid is found, the member is deleted.

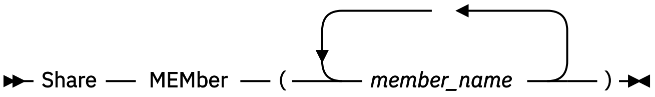
Message userid (or Message ALL)

Requests purging of a message member. Specify 'userid' to purge the message member MSG.userid for a specific user. Specify ALL to purge the common message member MSG.ALL (the period stands for X'FF').

Examples:

```
PURGE   LIBRARY(6) BEFORE(032883)
PURG    L(4) M(XYZ.ALL)
PUR     USER(S123) ALL
PURGE   COM MEM(RTN1)
PUR     BROAD
PURGE   MESSAGE S123
PURGE   MESSAGE ALL
```

SHARE Command



The SHARE command is used to flag members as common and thus make them available to all users. After the member (or members) has been made common, a directory entry for it is placed into each library's directory (except for libraries which were added with the NOCOMMON specification).

The SHARE command can be used only when VSE/ICCF is not active or when the VSE/ICCF library file is disconnected. Also, it must not be used for macros and procedures supplied with VSE/ICCF.

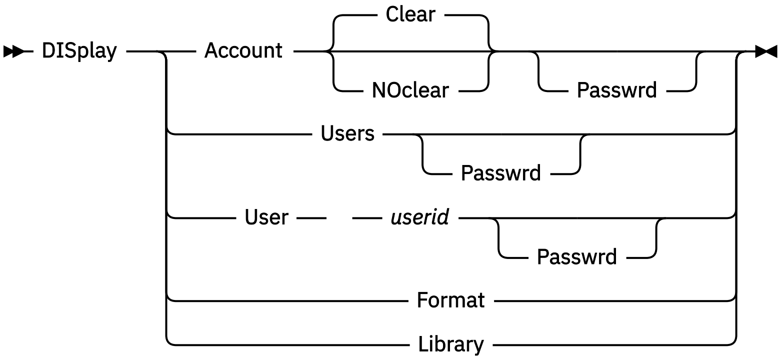
MEMber (member_name....
Specifies a member name or a list of member names. The member name(s) must be unique within the entire VSE/ICCF library file. If a member was password-protected, it will remain protected with the same password.

Examples:

```
SHARE  MEM(TESTDATA)
S      M(DATA1 DATA2 DATA4)
SHAR   MEM(COMPILE)
```

Display or Clear Accounting Information

DISPLAY Command



Account
Requests printing and punching of accumulated accounting information. The accounting information which is punched is an 80-column image of part of the user profile record as it appears on disk. The format of that record is described below.

Clear
NOclear
Specifies the clearing action to be taken. If omitted or specified as CLEAR, the accounting information is reset to zero. If specified as NOCLEAR, the information is left for further accumulation.

Passwrd
Causes a user's password to be displayed along with the rest of the profile information. This operand is only valid if either VSE/ICCF is not up or, if VSE/ICCF is up, the VSE/ICCF library file is disconnected.

Users
User userid
The USERS option causes a display of all user profile records to be printed along with the accounting information. It performs the same function as ACCOUNT but no clearing or punching takes place. If a four-character userid follows the keyword USER (or USERS), only that user's profile record is displayed.

DISPLAY USERS will function online and while VSE/ICCF is running, but DISPLAY ACCOUNT will not.

Format
Shows the number of libraries and user profiles specified.

Library
Normally, execution of the DISPLAY command also produces a summary of library file space usage at the end of the report. Issuing DISPLAY LIBRARY causes the display of only the library file space usage.

Three numbers are displayed:

- The number of records in the VSE/ICCF library file (X)

This is the maximum number of records which could possibly be allocated to user profiles, libraries, directory records, broadcast records, members or spool areas.

- The number of records in library members (Y)

  This is the approximate number of records which are chained from directory records for the listed user(s).

- The number of available high file records (Z)

  The number of high 4 for any purpose. The approximate number of records in the free chain may be calculated as

  ```
  X - Y - Z - R
  ```

  where R is the number of records associated with users whose profile records were not displayed.

After a selective restore (see "RESTORE (Member) Command" on page 141), the DISPLAY USER command will give incorrect file statistics.

The 80-column card image format of the accounting information is shown in Table 7 on page 147 .

Table 7. Card-Image Format of Accounting Information

Start Position	Length	Field Type	Description
1	2	Character	Always 'B$'
3 *	4	Character	Userid
7	2	Binary	Library assigned to user
9	6	Blanks	Reserved
15	2	Binary	Reserved
17 *	4	Binary	Number of logons
21 *	4	Binary	Number of ENTER key requests
25	4	Binary	Reserved
29 *	4	Binary	Number of execution requests
33 *	4	Binary	Total execution units used
37	4	Blanks	Reserved
41 *	8	Binary	Date of last logon, mm/dd/yy or dd/mm/yy
49 *	4	Binary	Total space usage
53 *	4	Binary	Total logon time
57 *	4	Binary	Total time in interactive partition
71 *	2	Binary	Century of date (19 or 20)

: * The field is related to accounting functions.

Following is a discussion of the fields that contain statistical information:

- Number of Logons

  The number of times this userid was logged on since the statistics fields were cleared last. This value is incremented by one each time a user logs on and successfully completes the userid validation. The logon is counted even if the user fails to complete the password validation and logs off. This value includes logons from a terminal as well as logons via the DTSBATCH utility program.

- Number of ENTER Key Requests

The number of inputs from this userid since the statistics fields were cleared last. This value is incremented by one each time input is passed to VSE/ICCF after logon has been completed. Input necessary to log on is not counted. This value includes input submitted from the terminal as well as input submitted through the DTSBATCH utility program.

- Number of Execution Requests

  The number of /RUN or /EXEC requests from this userid since the statistics fields were cleared last. This value is incremented by one each time a /RUN or /EXEC request is recognized. Even if the /RUN or /EXEC request cannot be processed (for example, if the user already has another job executing in an interactive partition), the count is still incremented. Only if the user is not eligible to execute jobs in interactive partitions (OPTA bit 4 of the user profile ON) is a /RUN or /EXEC request not counted. However, all submit-to-batch requests are included in this count regardless of the setting of the OPTA bit 4 switch.

- Total Execution Units Used

  The number of execution units of time this userid had jobs in an interactive partition eligible for execution since the statistics fields were cleared last. Each execution unit equals 1.049 seconds. This time does not necessarily represent actual processor time used by jobs in interactive partitions.

- Date of Last Logon

  The date of the last /LOGON command issued with this userid. A new date is kept even if the user subsequently fails to complete the password validation and logs off. When the user logs on, the date is copied from the communication region of the VSE partition in which VSE/ICCF is running. The format of the date depends on the format defined in the partition communication region.

- Total Space Usage

  The number of logical records in the VSE/ICCF library (DTSFILE) currently being used to hold library members for this userid. This count is a measure of how much disk space is required by this userid. It does not include library space for the user profile records.

- Total Logon Time

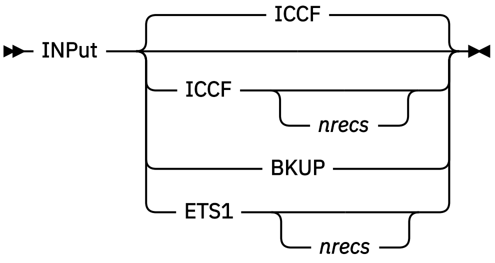
  The number of the time intervals, in seconds that this userid was logged on. The time of day is saved when the user completes the /LOGON command. If the user should fail to complete the password validation and log off, the elapsed time is still counted. At the time the user logs off, the difference between the /LOGON time and the current time of day is computed.

- Total Time in Interactive Partition

  The sum of the elapsed time intervals, in seconds, for all jobs submitted under this userid for interactive partition execution. The interactive partition start time is saved at the point in time when an interactive partition is allocated to a job. The interactive partition elapsed time is recomputed on every VSE/ICCF High Priority Task time interrupt so long as the job remains in the interactive partition. The elapsed time is computed by subtracting the start time from the current time of the interrupt (the resolution is to the nearest second).

Displaying and Punching Libraries, Members, Directories

INPUT Command



This command may be used online under VSE/ICCF or in another VSE partition while VSE/ICCF is running. If the operations to be performed by DTSUTIL include only DSERV, PRINT or PUNCH operations from the backup file, then the VSE/ICCF library file will not be opened and label information for the VSE/ICCF library file will not be required.

Commands which require the VSE/ICCF library file for input are not permitted while INPUT is from a backup file.

BKUP

Allows a backup file to be used as the source for the commands DSERV, PRINT, PUNCH and PRTPCH (do not use the INPUT BKUP format prior to any other command, such as RESTORE MEMBER). If INPUT BKUP is specified and, for example, the DSERV command follows, the directory listing displayed will be from the directories on the backup file DTSRSTR. To return DTSUTIL to normal operation, INPUT ICCF must be specified. The backup file is named DTSRSTR and must be assigned to SYS004 if on tape.

The ETSS II specifications for the backup file are still accepted, but BKUP is the preferred operand.

ICCF
ICCF nrecs

Requests resuming of normal DTSUTIL operation. If INPUT ICCF is followed by a decimal number 'nrecs', this number will be interpreted as the number of records per block in the VSE/ICCF library file. This must be less than or equal to the value of the NRECS tailoring option (in the DTSOPTNS tailoring macro). This facility allows DTSUTIL to be executed on VSE/ICCF library files of differing block sizes.

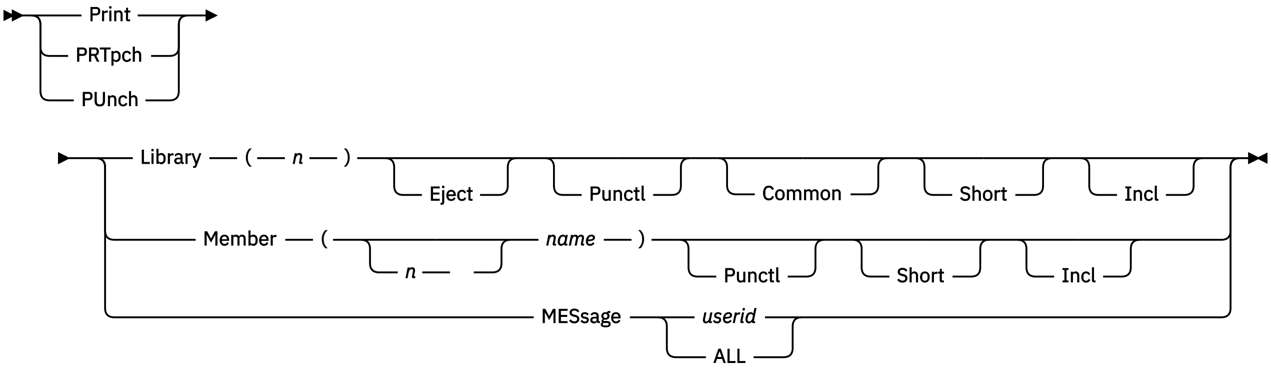
ETS1
ETS1 nrecs

Indicates that the library file to be used by DTSUTIL is one with the format from the original field developed program ETSS II. The 'nrecs' operand indicates the number of (88-byte) logical records in a block. This operand must be specified if the block size of the library file from ETSS II is smaller than the block size used with VSE/ICCF. If the block sizes are equal no operand is required. If the ETSS II block size is larger than that used in VSE/ICCF, this command may not be used (file error will result).

This last form of the INPUT command is used to transfer data from a library file with ETSS II format via the PRINT, PUNCH, PRTPCH, DSERV, and RESTORE DATAONLY commands. It is also possible to perform the RESTORE function to this ETSS II format library file. If RESTORE of the whole system is desired the option DATAONLY is assumed.

Note: Once you have started processing in either ICCF or ETS1 mode you may not change to any of the other modes within the same DTSUTIL execution.

PRINT and PUNCH Commands



The PRINT, PUNCH and PRTPCH commands provide for the displaying of VSE/ICCF libraries or members. This can be done from the VSE/ICCF library file or from the backup file (refer to the INPUT command for a further discussion). VSE/ICCF does not need to be down for these commands to be processed. Also, the commands can be entered for execution of DTSUTIL in an interactive partition.

PRINT and PUNCH do what their names imply. PRTPCH causes both printing and punching to take place; the minimum abbreviation is three characters. If SYSPCH is assigned to disk or tape, 81-character records are written out.

The PRINT and PRTPCH commands cause a character display of the specified library or member. If a hexadecimal display is desired, refer to the DTSANALS utility.

Processing of these three commands is interruptible as follows:

- When the utility runs under VSE/ICCF, the /ATTEN command (or PA1 key) terminates this processing.
- When the utility runs in a VSE partition, the MSG xx command terminates this processing and causes the next command to be read.

Note: Compressed members will not be printed. However, the PUNCH command will allow compressed members to be written to SYSPCH (still in compressed form). If you later on place this output in a VSE/ICCF library member, you must set the 'compressed flag' attribute by issuing the /PROTECT ... CPRS command. The /PROTECT command is described in the VSE/ICCF User's Guide.

Library(n)
Specifies the number of the library whose members are to be displayed. All members of the library except the common members are displayed. If the library contains DBCS members and if the appropriate UPSI bit has been set (see “DTSUTIL UPSI Settings” on page 124), DTSUTIL prepares these members for printing on an IBM 3200 printer (with utility 5799-BGF, for example).

Member(name)
Member(n name)
Specifies the name of the member to be displayed and optionally the library in which it resides. If the library number is omitted, the input file is searched beginning with the first library, until a match is found. If the specified member is a DBCS member and if the appropriate UPSI bit has been set (see“DTSUTIL UPSI Settings” on page 124), DTSUTIL prepares this member for printing on an IBM 3200 printer (with utility 5799-BGF, for example).

MESsage userid
MESsage ALL
Requests printing or punching of a message member. Specify 'userid' to print/punch the message member MSG.userid for a specific user. Specify ALL to print/punch the member for common messages, MSG.ALL (the period stands for x'FF').

Eject
Specifies that each member is to be listed starting at the top of a page. It has no effect if MEMBER or PUNCH is specified. Printing of a single member always begins on a new page unless the save-paper option is in effect (refer to “DTSUTIL UPSI Settings” on page 124 for more information). If this option is omitted, three spaces and a scale line will be printed between each member.

Punctl
Specifies that ADD MEMBER and END OF MEMBER cards are to be punched for each member. This option is useful if the output is to be used for DTSUTIL input to another file. PUNCTL should not be specified on the PRINT command. If this option is omitted, no control cards are punched.

If PUNCTL is specified, statements beginning with /* or /& appear as statements beginning with ../* or ../&, respectively; VSE/POWER JECL statements, which begin with * $$, appear as statements beginning with ..$$, so that they can be processed by DTSUTIL.

If you punch and add members using this option, these members become public.

Common
Specifies that members flagged as common are to be displayed. A member requested by 'name' will be displayed even if common. Also, a member included due to a /INCLUDE is displayed, even if common. If this operand is omitted, common members will not be displayed.

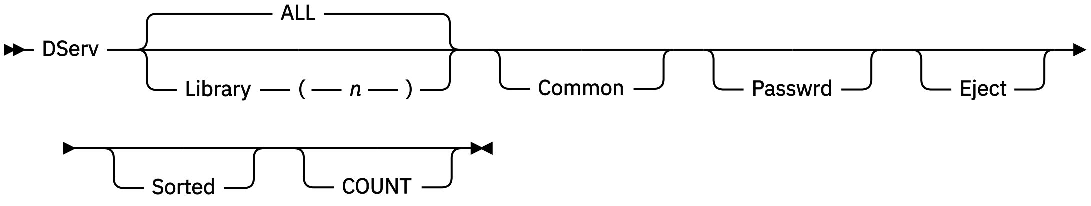
Short
Specifies that only the first 10 lines of a member are to be listed or punched. This may be useful in trying to determine what various members consist of.

Incl

Requests the inclusion of members that are specified in /INCLUDE statements within the member to be printed/punched. INCL is not allowed if input is a backup file. /INCLUDE statements within included members (nested /INCLUDEs) are resolved down to a depth of eight. 256 is the maximum of /INCLUDE statements that can be processed with one command.

/INCLUDE statements in compressed members will not be detected and, therefore, not be resolved. The search for an included member starts in the library where the 'invoking' member resides. If the member is not found here, the VSE/ICCF library file is searched, beginning with library 1, until the member is found.

DSERV Command



The DSERV command prints the directory listing of all the members in a library or in the VSE/ICCF library file. In addition, the attributes of the library are also displayed.

Library(n)
ALL

With ALL specified or assumed as default, the directory listing will encompass the entire VSE/ICCF library file. Library(n) requests limit the listing to a certain library.

Common

Indicates that the names of members flagged as common will be printed in addition to the other members in the libraries. If omitted, common member names will not be printed.

Passwrd

Indicates that the passwords of the members will be printed. This operand is only valid if either VSE/ICCF is not up or, if VSE/ICCF is up, the VSE/ICCF library file is disconnected.

Eject

Indicates that the directory for each library will begin on a new page. If omitted, about four lines are skipped before beginning the next directory listing. This option is useful if the directory listings are going to be split apart for users of the various libraries.

Sorted

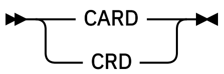
Indicates that the directory should be printed in alphabetical order.

COUNT

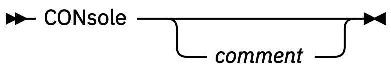
When the COUNT operand is specified, the DSERV listing will show the number of records per member. The number will be in the same position as in the listing for the BACKUP command. COUNT is ignored when DSERV PASSWRD operand is specified, because output for both operands would be at the same position. DSERV COUNT against a backup file via INPUT BKUP command is not possible.

Since VSE/ICCF 1.3.0, the date of the last replace operation for a member is internally stored in the format month/day/year. The DSERV output shows this date in the format defined in the system communication region (MDY or DMY). For library members created before VSE/ICCF 1.3.0, the date is stored internally in either of the formats DMY or MDY, depending on the setting in the communication region. To avoid misleading displays, the date can be set by the DATE option of the /PROTECT command. Refer to " /PROTECT" in the VSE/ICCF User's Guide.

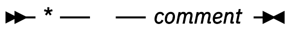
Miscellaneous Commands



When entered from the console, this command causes all subsequent commands to be read from SYSIPT. When entered from SYSIPT, it is ignored.

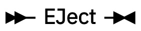


When read from the SYSIPT input stream, this command causes all subsequent commands to be read from the console. If a comment operand is specified, the comment will appear on the ENTER DTSUTIL command prompt line.

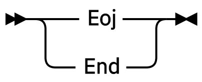


When read from either the console or the input stream, this command causes the comment following the asterisk to appear on the output listing. The comment is not examined for operands or commands.

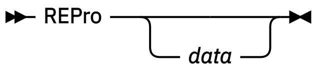
An * command should not follow broadcast records to signal their end. VSE/ICCF interprets the command as a broadcast record.



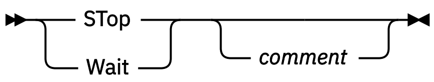
This command causes an immediate skip to head of forms and a new heading to be printed. The command itself does not appear on the output listing. It overrides the paper-save option (see “DTSUTIL UPSI Settings” on page 124, bit h, for more information about this option).



These commands all indicate processing is complete. All files are closed and the job is ended. However, /* on SYSIPT indicates end-of-file and terminates DTSUTIL.

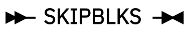


This command causes the text following the command to be punched into a card. If no data appears on the line with the command, the next line of input is used as the text. This command can be useful for punching a member to tape for later processing by some program which requires control card separators.



These commands cause DTSUTIL to wait for an operator communication interrupt. They are usually used during console operations to free the console for messages from other partitions. MSG xx will cause the processing of commands to resume ('xx' is the partition where DTSUTIL is running).

When the STOP/WAIT command is entered from SYSIPT, the command along with any comment operand is logged on the console.



This command requests that DTSUTIL command processing ignores blank records. After this command has been issued, DTSUTIL will not treat a blank record as an invalid command.

DTSUTIL Utility Examples

The examples below assume that a **DLBL/EXTENT set for DTSFILE** is available in the VSE standard label area. An assignment for the VSE/ICCF library file (via procedure DTRICCF) is shown, but you may just as well have a permanent assignment for DTSFILE.

Do not specify SIZE=AUTO in the EXEC DTSUTIL statement (otherwise you might get message K242I).

Example 1: This example may also be used to create the initial archive file.

```
// JOB      BACK UP THE LIBRARY FILE TO TAPE
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// TLBL     DTSBKUP,'T/S BACKUP',99/365
// ASSGN    SYS005,cuu          (tape drive)
// EXEC     DTSUTIL
BACKUP
/*
/&
```

Example 2:

```
// JOB      BACK UP THE LIBRARY FILE TO DISK
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// DLBL     DTSBKUP,'T/S BACKUP 0',99/365
// EXTENT   SYS012,vol-id,1,0,start,length
// ASSGN    SYS012,cuu     (disk drive)
// EXEC     DTSUTIL
BACKUP
/*
/&
```

Example 3:

```
// JOB     SELECTIVE RESTORE FROM TAPE
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// TLBL    DTSRSTR,'T/S BACKUP'
// ASSGN   SYS004,cuu     (tape drive)
// EXEC    DTSUTIL
RESTORE    MEMBER(8 OLDMEM)
/*
/&
```

Example 4:

```
// JOB     DELETE STUDENT MEMBERS
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC    DTSUTIL
DELETE  USER (STUA)     (deletes userid)
DELETE  USER (STUB)     (deletes userid)
PURGE   LIB(16) ALL     (leaves common and library header
/*                      record intact)
/&
```

Example 5:

```
// JOB     PURGE ALL MEMBERS OF A SPECIFIC USER
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC    DTSUTIL
PURGE   USER(MANA) ALL
/*
/&
```

Example 6:

```
// JOB     SAVE ACCOUNTING DATA AND CLEAR
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
ASSGN    SYSPCH,cuu    (card, tape or disk)
// EXEC    DTSUTIL
DISPLAY    ACCOUNT CLEAR
```

```
/*
/&
```

Example 7:

```
// JOB     CREATE NEW ARCHIVE FILE
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// ASSGN   SYS004,cuu      (tape input)
// TLBL    DTSMERG,'T/S BACKUP 0',99/365
// ASSGN   SYS005,cuu      (tape output)
// TLBL    DTSBKUP,'T/S BACKUP 0'
// EXEC    DTSUTIL
MERGE
END
/*
/&
```

Example 8:

```
// JOB    PRINT A MEMBER
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC   DTSUTIL
PR  MEM(4 ANYMEM)
END
/&
```

Example 9:

```
// JOB     PRINT AND PUNCH A LIBRARY INCLUDING COMMON
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC    DTSUTIL
PRTPCH     LIB(4) COMMON
/*
/&
```

Example 10:

```
// JOB     DISPLAY DIRECTORIES
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC    DTSUTIL
DSERV      ALL COM
END
/&
```

Example 11:

```
// JOB    PUNCH MEMBER AND CHANGE NAME FOR ANOTHER VSE/ICCF
*         INSTALLATION
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC   DTSUTIL
* REPRODUCE ADD MEMBER FOR DIFFERENT NAME AND LIB
REPRO ADD MEMBER (6 NEWNAME USRA PASW)
* PUNCH MEMBER
PUNCH MEMBER (12 OLDNAME)
* REPRODUCE END OF MEMBER CARD
REPRO
END OF MEMBER
/*
/&
```

Example 12:

```
// JOB   INVOKE DTSUTIL FROM CARDS FOR CONSOLE INPUT
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC  DTSUTIL
CONSOLE
/&
```

Example 13:

```
// JOB   FORMAT AND RESTORE VSE/ICCF LIBRARY FROM TAPE
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// TLBL  DTSRSTR,'ICCF BACKUP 1'
// ASSGN SYS004,cuu                (tape drive)
// EXEC  DTSUTIL
FORMAT  LIB(21) USERS(57)
RESTORE
ADD  BROADCAST
* first broadcast record
* second broadcast record
/*
/&
```

Example 14:

```
// JOB   SET UP VSE/ICCF LIBRARY FILE INITIALLY
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC  DTSUTIL
FORMAT   LIB(30) USER(64)
ADD  LIB FREE(25)              (formats first library record)
ADD  LIB MAX (200) FREE(25)    (formats second library record)
ADD  LIB MAX (200) FREE(25)    (formats third library record)
ADD  LIB MAX (64)  FREE(25)    (formats fourth library record)
ADD  USER(AAAA) LIB(1) PASS(A$ADMN) OPTA(01110111) OPTB(11111110)
ADD  USER(USRB) LIB(4) PASS(PASWDB) MAXSTATE(350)
ADD  USER(USRC) LIB(4) PASS(PASWDC) MAXSTATE(400)
ADD  USER(APCD) LIB(3) PASSWRD(POLUTN) OPTA(11011001)
ADD  USER(APCD) LIB(3) PASSWRD(POLUTN) MAXST(850)
ADD  BROADCAST
* first broadcast record
* second broadcast record
ADD  MEMBER (2 COMRTN1 APCD)
 ...
 ...                          (cards of member)
 ...
END OF MEMBER
/*
/&
```

Example 15:

```
// JOB    DISPLAY MEMBER FROM BACKUP FILE & FROM VSE/ICCF
*         LIBRARY
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// DLBL   DTSRSTR,'ICCF BACKUP'
// EXTENT SYSnnn,vol-id.
// ASSGN  SYSnnn,cuu     (disk drive)
// EXEC   DTSUTIL
INPUT BKUP
PRINT MEM (SOMEMEM)
INPUT ICCF
PRINT MEM (SOMEMEM)
END
/&
```

Example 16:

```
// JOB   LIST MEMBERS ON BACKUP FILE
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// TLBL  DTSRSTR,'ICCF BACKUP'
// ASSGN SYS004,cuu   (tape drive)
// EXEC  DTSUTIL
INPUT BKUP
DSERV ALL COMMON
/*
/&
```

Example 17:

```
// JOB   SELBACK SELECTIVE BACKUP/RESTORE
*                BACK UP A PARTICULAR LIBRARY
*                THIS JOB PUNCHES ALL MEMBERS OF LIBRARY 12 TO
*                TAPE, INCLUDING THE ADD MEM (12) membername
*                CONTROL CARDS
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// ASSGN SYSPCH,280     (tape drive)
// MTC   REW,280
// EXEC  DTSUTIL
PUNCH LIBRARY(12) PUNCTL
/*
// MTC WTM,SYSPCH,2
// MTC REW,SYSPCH
/&
// JOB   SELREST RESTORE ALL MEMBERS OF LIBRARY 12
*  1.STEP:  DELETE ALL MEMBERS FROM LIBRARY 12
// EXEC PROC=DTRICCF            (assign VSE/ICCF library file)
// EXEC DTSUTIL
PURGE LIBRARY(12) ALL
/*
*  2.STEP:  ADD ALL MEMBERS FROM TAPE
// ASSGN SYSIPT,280     (tape drive)
// MTC   REW,280
// EXEC  DTSUTIL
/&
```

Appendix A. Understanding Syntax Diagrams

This section describes how to read the syntax diagrams.

To read a syntax diagram follow the path of the line. Read from left to right and top to bottom.

- The ▶▶—— symbol indicates the beginning of a syntax diagram.
- The ——▶ symbol, at the end of a line, indicates that the syntax diagram continues on the next line.
- The ▶—— symbol, at the beginning of a line, indicates that a syntax diagram continues from the previous line.
- The ——▶◀ symbol indicates the end of a syntax diagram.

Syntax items (for example, a keyword or variable) can be:

- Directly on the line (required)
- Above the line (default)
- Below the line (optional)

Uppercase Letters

Uppercase letters denote the shortest possible abbreviation. If an item appears entirely in uppercase letters, it cannot be abbreviated.

You can type the item in uppercase letters, lowercase letters, or any combination. For example:

▶▶— KEYWOrd —▶◀

In this example, you can enter KEYWO, KEYWOR, or KEYWORD in any combination of uppercase and lowercase letters.

Symbols

You must code these symbols exactly as they appear in the syntax diagram.

*****

Asterisk

:

Colon

,

Comma

=

Equal sign

-

Hyphen

//

Double slash

()

Parenthesis

.

Period

+

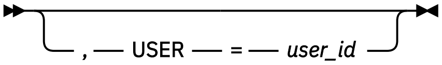
Add

For example:

```
* $$ LST
```

Variables

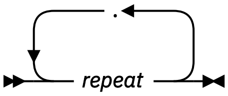
Highlighted lowercase letters denote variable information that you must substitute with specific information. For example:



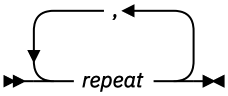
Here you must code USER= as shown and supply an ID for user_id. You can enter USER in lowercase, but you should not change it otherwise.

Repetition

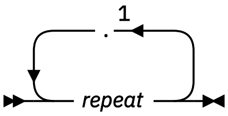
An arrow returning to the left means that the item can be repeated.



A character within the arrow means you must separate repeated items with that character.



A footnote (1) by the arrow references a limit that tells how many times the item can be repeated.

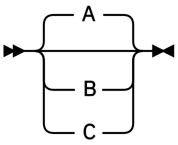


Notes:

[1] Specify *repeat* up to five times.

Defaults

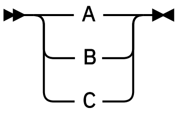
Defaults are above the line. The system uses the default unless you override it. You can override the default by coding an option from the stack below the line. For example:



In this example, A is the default. You can override A by choosing B or C.

Required Choices

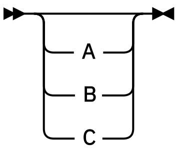
When two or more items are in a stack and one of them is on the line, you must specify one item. For example:



Here you must enter either A or B or C.

Optional Choice

When an item is below the line, the item is optional. You can only choose one item. For example:



Here you can enter either A or B or C, or omit the field.

Required Blank Space

A required blank space is indicated as such in the notation. For example:

```
* $$ EOJ
```

This indicates that at least one blank is required before and after the characters $$.

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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

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

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

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

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

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

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

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

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

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

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

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

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

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

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

COPYRIGHT LICENSE:

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

Programming Interface Information

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain services of z/VSE.

Trademarks

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

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

IPv6/VSE is a registered trademark of Barnard Software, Inc.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

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

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

Terms and Conditions for Product Documentation

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

Applicability

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

Personal use

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

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein. IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed. You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

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

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/VSE enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using Assistive Technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/VSE. Consult the assistive technology documentation for specific information when using such products to access z/VSE interfaces.

Documentation Format

The publications for this product are in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when you use the PDF files and want to request a web-based format for a publication, you can either write an email to s390id@de.ibm.com, or use the Reader Comment Form in the back of this publication or direct your mail to the following address:

```
IBM Deutschland Research & Development GmbH
Department 3282
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany
```

In the request, be sure to include the publication number and title.

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

Glossary

This glossary includes terms and definitions for IBM z/VSE.

The following cross-references are used in this glossary:

1. See refers the reader from a term to a preferred synonym, or from an acronym or abbreviation to the defined full form.
2. See also refers the reader to a related or contrasting term.

A

Access Control Logging and Reporting

An IBM licensed program to log all attempts of access to protected data and to print selected formatted reports on such attempts.

access control table (DTSECTAB)

A table that is used by the system to verify a user's right to access a certain resource.

access list

A table in which each entry specifies an address space or data space that a program can reference.

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 VTAM.

account file

A disk file that is maintained by VSE/POWER containing accounting information that is generated by VSE/POWER and the programs running under VSE/POWER.

addressing mode (AMODE)

A program attribute that refers to the address length that a program is prepared to handle on entry. Addresses can be either 24 bits, 31 bits, or 64 bits in length. In 24 bit addressing mode, the processor treats all virtual addresses as 24-bit values; in 31 bit addressing mode, the processor treats all virtual addresses as 31-bit values and in 64-bit addressing mode, the processor treats all virtual addresses as 64-bit values. Programs with an addressing mode of ANY can receive control in either 24 bit or 31 bit addressing mode. 64 bit addressing mode cannot be used as program attribute.

administration console

In z/VSE, one or more consoles that receive all system messages, except for those that are directed to one particular console. Contrast this with the user console, which receives only those messages that are directed to it, for example messages that are issued from a job that was submitted with the request to echo its messages to that console. The operator of an administration console can reply to all outstanding messages and enter all system commands.

alternate block

On an FBA disk, a block that is designated to contain data in place of a defective block.

alternate index

In systems with VSE/VSAM, the index entries of a given base cluster that is organized by an alternate key, that is, a key other than the prime key of the base cluster. For example, a personnel file preliminary ordered by names can be indexed also by department number.

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.

alternate track

A library, which becomes accessible from a terminal when the user of that terminal issues a connect or switch (library) request.

AMODE

Addressing mode.

APA

All points addressable.

APAR

Authorized Program Analysis Report.

appendage routine

A piece of code that is physically located in a program or subsystem, but logically and extension of a supervisor routine.

application profile

A control block in which the system stores the characteristics of one or more application programs.

application program

A program that is written for or by a user that applies directly to the user's work, such as a program that does inventory control or payroll. See also batch program and online application program.

AR/GPR

Access register and general-purpose register pair.

ASC mode

Address space control mode.

ASI (automated system initialization) procedure

A set of control statements, which specifies values for an automatic system initialization.

attention routine (AR)

A routine of the system that receives control when the operator presses the Attention key. The routine sets up the console for the input of a command, reads the command, and initiates the system service that is requested by the command.

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 VSE/POWER with little or no operator involvement.

auxiliary storage

Addressable storage that is not part of the processor, for example storage on a disk unit. Synonymous with external storage.

B

B-transient

A phase with a name beginning with $$B and running in the Logical Transient Area (LTA). Such a phase is activated by special supervisor calls.

bar

2 GigyByte (GB) line

basic telecommunications access method (BTAM)

An access method that permits read and write communication with remote devices. BTAM is not supported on z/VSE.

BIG-DASD

A subtype of Large DASD that has a capacity of more than 64 K tracks and uses up to 10017 cylinders of the disk.

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 a control block.

block group

In VSE/POWER, the basic organizational unit for fixed-block architecture (FBA) devices. Each block group consists of a number of 'units of transfer' or blocks.

C

CA splitting

Is the host part of the VSE JavaBeans, and is started using the job STARTVCS, which is placed in the reader queue during installation of z/VSE. Runs by default in dynamic class R. In VSE/VSAM, to double a control area dynamically and distribute its CIs evenly when the specified minimum of free space get used up by more data.

carriage control character

The fist character of an output record (line) that is to be printed; it determines how many lines should be skipped before the next line is printed.

catalog

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. To store a library member such as a phase, module, or book in a sublibrary. See also VSE/VSAM catalog.

cell pool

An area of virtual storage that is obtained by an application program and managed by the callable cell pool services. A cell pool is located in an address space or a data space and contains an anchor, at least one extent, and any number of cells of the same size.

central location

The place at which a computer system's control device, normally the systems console in the computer room, is installed.

chained sublibraries

A facility that allows sublibraries to be chained by specifying the sequence in which they must be searched for a certain library member.

chaining

A logical connection of sublibraries to be searched by the system for members of the same type (phases or object modules, for example).

channel command word (CCW)

A doubleword at the location in main storage that is specified by the channel address word. One or more CCWs make up the channel program that directs data channel operations.

channel program

One or more channel command words that control a sequence of data channel operations. Execution of this sequence is initiated by a start subchannel instruction.

channel scheduler

The part of the supervisor that controls all input/output operations.

channel subsystem

A feature of z/Architecture that provides extensive additional channel (I/O) capabilities to IBM Z.

channel to channel attachment (CTCA)

A function that allows data to be exchanged

1. Under the control of VSE/POWER between two virtual VSE machines running under VM or
2. Under the control of VTAM between two processors.

character-coded request

A request that is encoded and transmitted as a character string. Contrast with *field-formatted request*.

checkpoint

1. A point at which information about the status of a job and the system can be recorded so that the job step can be restarted later.
2. To record such information.

CICS (Customer Information Control System)

An IBM program that controls online communication between terminal users and a database. Transactions that are entered at remote terminals are processed concurrently by user-written application programs. The program includes facilities for building, using, and servicing databases.

CICS ECI

The CICS External Call Interface (ECI) is one possible requester type of the *CICS business logic interface* that is provided by the CICS Transaction Server for z/VSE. It is part of the CICS client and allows workstation programs to CICS function on the z/VSE host.

CICS EXCI

The EXternal CICS Interface (EXCI) is one possible requester type of the *CICS business logic interface* that is provided by the CICS Transaction Server for z/VSE. It allows any BSE batch application to call CICS functions.

CICS system definition data set (CSD)

A VSAM KSDS cluster that contains a resource definition record for every record defined to CICS using resource definition online (RDO).

CICS Transaction Server for z/VSE

A z/VSE base program that controls online communication between terminal users and a database. This is the successor system to CICS/VSE.

CICS TS

CICS Transaction Server

CICS/VSE

Customer Information Control System/VSE. No longer shipped on the Extended Base Tape and no longer supported, cannot run on z/VSE 5.1 or later.

class

In VSE/POWER, a group of jobs that either come from the same input device or go to the same output device.

CMS

Conversational monitor system running on z/VM.

common library

A library that can be interactively accessed by any user of the (sub)system that owns the library.

communication adapter

A circuit card with associated software that enables a processor, controller, or other device to be connected to a network.

communication region

An area of the supervisor that is set aside for transfer of information within and between programs.

component

1. Hardware or software that is part of a computer system.
2. A functional part of a product, which is identified by a component identifier.
3. In z/VSE, a component program such as VSE/POWER or VTAM.
4. In VSE/VSAM, a named, cataloged group of stored records, such as the data component or index component of a key-sequenced file or alternate index.

component identifier

A 12-byte alphanumeric string, uniquely defining a component to MSHP.

conditional job control

The capability of the job control program to process or to skip one or more statements that are based on a condition that is tested by the program.

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.

connection pooling

Introduced with an z/VSE 5.1 update to manage (reuse) connections of the z/VSE database connector in CICS TS.

connector

In the context of z/VSE, a connector provides the middleware to connect two platforms: Web Client and z/VSE host, middle-tier and z/VSE host, or Web Client and middle-tier.

connector (e-business connector)

A piece of software that is provided to connect to heterogeneous environments. Most connectors communicate to non-z/VSE Java-capable platforms.

container

Is part of the JVM of application servers such as the IBM WebSphere Application Server, and facilitates the implementation of servlets, EJBs, and JSPs, by providing resource and transaction management resources. For example, an EJB developer must not code against the JVM of the application server, but instead against the interface that is provided by the container. The main role of a container is to act as an intermediary between EJBs and clients, Is the host part of the VSE JavaBeans, and is started using the job STARTVCS, which is placed in the reader queue during the installation of z/VSE. Runs by default in dynamic class R. and also to manage multiple EJB instances. After EJBs have been written, they must be stored in a container residing on an application server. The container then manages all threading and client-interactions with the EJBs, and co-ordinate connection- and instance pooling.

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.

control program

A program to schedule and supervise the running of programs in a system.

conversational monitor system (CMS)

A virtual machine operating system that provides general interactive time sharing, problem solving, and program development capabilities and operates under the control of z/VM.

count-key-data (CKD) device

A disk device that store data in the record format: count field, key field, data field. The count field contains, among others, the address of the record in the format: cylinder, head (track), record number, and the length of the data field. The key field, if present, contains the record's key or search argument. CKD disk space is allocated by tracks and cylinders. Contrast with *FBA disk device. See also extended count-key-data device.*

cross-partition communication control

A facility that enables VSE subsystems and user programs to communicate with each other; for example, with VSE/POWER.

cryptographic token

Usually referred to simply as a *token*, this is a device, which provides an interface for performing cryptographic functions like generating digital signatures or encrypting data.

cryptography

1. A method for protecting information by transforming it (encrypting it) into an unreadable format, called ciphertext. Only users who possess a secret key can decipher (or decrypt) the message into plaintext.
2. The transformation of data to conceal its information content and to prevent its unauthorized use or undetected modification .

D

data block group

The smallest unit of space that can be allocated to a VSE/POWER job on the data file. This allocation is independent of any device characteristics.

data conversion descriptor file (DCDF)

With a DCDF, you can convert individual fields within a record during data transfer between a PC and its host. The DCDF defines the record fields of a particular file for both, the PC and the host environment.

data import

The process of reformatting data that was used under one operating system such that it can subsequently be used under a different operating system.

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 Language/I (DL/I)

A database access language that is used with CICS.

data link

In SNA, the combination of the link connection and the link stations joining network noes, for example, a z/Architecture channel and its associated protocols. A link is both logical and physical.

data security

The protection of data against unauthorized disclosure, transfer, modification, or destruction, whether accidental or intentional .

data set header record

In VSE/POWER abbreviated as DSHR, alias NDH or DSH. An NJE control record either preceding output data or, in the middle of input data, indicating a change in the data format.

data space

A range of up to 2 gigabytes of contiguous virtual storage addresses that a program can directly manipulate through z/Architecture instructions. Unlike an address space, a data space can hold only user data; it does not contain shared areas, or programs. Instructions do not execute in a data space. Contrast with address space.

data terminal equipment (DTE)

In SNA, the part of a data station that serves a data source, data sink, or both.

database connector

Is a function introduced with z/VSE 5.1.1, which consists of a client and server part. The client provides an API (CBCLI) to be used by applications on z/VSE, the server on any Java capable platform connects a JDBC driver that is provided by the database. Both client and server communicate via TCP/IP.

Database 2 (Db2)

An IBM rational database management system.

Db2-based connector

Is a feature introduced with VSE/ESA 2.5, which includes a customized Db2 version, together with VSAM and DL/I functionality, to provide access to Db2, VSAM, and DL/I data, using Db2 Stored Procedures.

Db2 Runtime only Client edition

The Client Edition for z/VSE comes with some enhanced features and improved performance to integrate z/VSE and Linux on z Systems.

Db2 Stored Procedure

In the context of z/VSE, a Db2 Stored Procedure is a Language Environment (LE) program that accesses Db2 data. However, from VSE/ESA 2.5 onwards you can also access VSAM and DL/I data using a Db2 Stored Procedure. In this way, it is possible to exchange data between VSAM and Db2.

DBLK

Data block.

DCDF

Data conversion descriptor file.

deblocking

The process of making each record of a block available for processing.

dedicated (disk) device

A device that cannot be shared among users.

device address

1. The identification of an input/output device by its device number.
2. In data communication, the identification of any device to which data can be sent or from which data can be received.

device driving system (DDS)

A software system external to VSE/POWER, such as a CICS spooler or PSF, that writes spooled output to a destination device.

Device Support Facilities (DSF)

An IBM supplied system control program for performing operations on disk volumes so that they can be accessed by IBM and user programs. Examples of these operations are initializing a disk volume and assigning an alternative track.

device type code

The four- or five-digit code that is used for defining an I/O device to a computer system. See also ICKDSF

dialog

In an interactive system, a series of related inquiries and responses similar to a conversation between two people. For z/VSE, a set of panels that can be used to complete a specific task; for example, defining a file.

dialog manager

The program component of z/VSE that provides for ease of communication between user and system.

digital signature

In computer security, encrypted data, which is appended to or part of a message, that enables a recipient to prove the identity of the sender.

Digital Signature Algorithm (DSA)

The Digital Signature Algorithm is the US government-defined standard for digital signatures. The DSA digital signature is a pair of large numbers, computed using a set of rules (that is, the DSA) and a set of parameters such that the identity of the signatory and integrity of the data can be verified. The DSA provides the capability to generate and verify signatures.

directory

In z/VSE the index for the program libraries.

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*.

disk operating system residence volume (DOSRES)

The disk volume on which the system sublibrary IJSYSRS.SYSLIB is located including the programs and procedures that are required for system startup.

disk sharing

An option that lets independent computer systems uses common data on shared disk devices.

disposition

A means of indicating to VSE/POWER how a job input or output entry is to be handled: according to its local disposition in the RDR/LST/PUN queue or its transmission disposition when residing in the XMT queue. A job might, for example, be deleted or kept after processing.

distribution tape

A magnetic tape that contains, for example, a preconfigured operating system like z/VSE. This tape is shipped to the customer for program installation.

DITTO/ESA for VSE

Data Interfile Transfer, Testing, and Operations utility. An IBM program that provides file-to-file services for disk, tape, and card devices.

DSF

Device Support Facilities.

DSH (R)

Data set header record.

dummy device

A device address with no real I/O device behind it. Input and output for that device address are spooled on disk.

duplex

Pertaining to communication in which data can be sent and received at the same time.

DU-AL (dispatchable unit - access list)

The access list that is associated with a z/VSE main task or subtask. A program uses the DU-AL associated with its task and the PASN-AL associated with its partition. See also “PASN-AL (primary address space number - access list)” on page 188.

dynamic class table

Defines the characteristics of dynamic partitions.

dynamic partition

A partition that is created and activated on an 'as needed' basis that does not use fixed static allocations. After processing, the occupied space is released. Dynamic partitions are grouped by class, and jobs are scheduled by class. Contrast with *static partition*.

dynamic space reclamation

A librarian function that provides for space that is freed by the deletion of a library member to become reusable automatically.

E

ECI

See "CICS ECI" on page 171.

emulation

The use of programming techniques and special machine features that permit a computer system to execute programs that are written for another system or for the use of I/O devices different from those that are available.

emulation program (EP)

An IBM control program that allows a channel-attached 3705 or 3725 communication controller to emulate the functions of an IBM 2701 Data Adapter Unit, or an IBM 2703 Transmission Control.

end user

1. A person who makes use of an application program.
2. In SNA, the ultimate source or destination of user data flowing through an SNA network. Might be an application program or a terminal operator.

Enterprise Java Bean

An EJB is a distributed bean. "Distributed" means, that one part of an EJB runs inside the JVM of a web application server, while the other part runs inside the JVM of a web browser. An EJB either represents one data row in a database (entity bean), or a connection to a remote database (session bean). Normally, both types of an EJB work together. This allows to represent and access data in a standardized way in heterogeneous environments with relational and non-relational data. See also *JavaBean*.

entry-sequenced file

A VSE/VSAM file whose records are loaded without respect to their contents and whose relative byte addresses cannot change. Records are retrieved and stored by addressed access, and new records are added to the end of the file.

Environmental Record Editing and Printing (EREP) program

A z/VSE base program that makes the data that is contained in the system record file available for further analysis.

EPI

See *CICS EPI.*

ESCON Channel (Enterprise Systems Connection Channel)

A serial channel, using fiber optic cabling, that provides a high-speed connection between host and control units for I/O devices. It complies with the ESA/390 and IBM Z I/O Interface until z114. The zEC12 processors do not support ESCON channels.

exit routine

1. Either of two types of routines: installation exit routines or user exit routines. Synonymous with exit program.
2. See *user exit routine*.

extended addressability

The ability of a program to use 31 bit or 64 bit virtual storage in its address space or outside the address space.

extended recovery facility (XRF)

In z/VSE, a feature of CICS that provides for enhanced availability of CICS by offering one CICS system as a backup of another.

External Security Manager (ESM)

A priced vendor product that can provide extended functionality and flexibility that is compared to that of the Basic Security Manager (BSM), which is part of z/VSE.

F

FASTCOPY

See "VSE/Fast Copy" on page 199.

fast copy data set program (VSE/Fast Copy)

See "VSE/Fast Copy" on page 199.

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.

FAT-DASD

A subtype of Large DASD, it supports a device with more than 4369 cylinders (64 K tracks) up to 64 K cylinders.

FCOPY

See *VSE/Fast Copy.*

fence

A separation of one or more components or elements from the remainder of a processor complex. The separation is by logical boundaries. It allows simultaneous user operations and maintenance procedures.

fetch

1. To locate and load a quantity of data from storage.

2. To bring a program phase into virtual storage from a sublibrary and pass control to this phase.
3. The name of the macro instruction (FETCH) used to accomplish 2. See also *loader.*

Fibre Channel Protocol (FCP)

A combination of hardware and software conforming to the Fibre Channel standards and allowing system and peripheral connections via FICON and FICON Express feature cards on IBM zSeries processors. In z/VSE, zSeries FCP is employed to access industry-standard SCSI disk devices.

fragmentation (of storage)

Inability to allocate unused sections (fragments) of storage in the real or virtual address range of virtual storage.

FSU

Fast service upgrade.

FULIST (FUnction LIST)

A type of selection panel that displays a set of files and/or functions for the choice of the user.

G

generation

See *macro generation.*

generation feature

An IBM licensed program order option that is used to tailer the object code of a program to user requirements.

GETVIS space

Storage space within partition or the shared virtual area, available for dynamic allocation to programs.

guest system

A data processing system that runs under control of another (host) system. On the mainframe z/VSE can run as a guest of z/VM.

H

hard wait

The condition of a processor when all operations are suspended. System recovery from a hard wait is impossible without performing a new system startup.

hash function

A hash function is a transformation that takes a variable-size input and returns a fixed-size string, which is called the hash value. In cryptography, the hash functions should have some additional properties:

- The hash function should be easy to compute.
- The hash function is one way; that is, it is impossible to calculate the 'inverse' function.

- The hash function is collision-free; that is, it is impossible that different input leads to the same hash value.

hash value

The fixed-sized string resulting after applying a *hash function* to a text.

High-Level Assembler for VSE

A programming language providing enhanced assembler programming support. It is a base program of z/VSE.

home interface

Provides the methods to instantiate a new EJB object, introspect an EJB, and remove an EJB instantiation., as for the remote interface is needed because the deployment tool generates the implementation class. Every Session bean's home interface must supply at least one *create()* method.

host mode

In this operating mode, a PC can access a VSE host. For programmable workstation (PWS) functions, the Move Utilities of VSE can be used.

host system

The controlling or highest level system in a data communication configuration.

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.

HTTP Session

In the context of z/VSE, identifies the web-browser client that calls a servlet (in other words, identifies the connection between the client and the middle-tier platform).

I

ICCF

See *VSE/ICCF.*

ICKDSF (Device Support Facilities)

A z/VSE base program that supports the installation, use, and maintenance of IBM disk devices.

include function

Retrieves a library member for inclusion in program input.

index

1. A table that is used to locate records in an indexed sequential data set or on indexed file.
2. In, an ordered collection of pairs, each consisting of a key and a pointer, used by to sequence and locate the records of a key-sequenced data set or file; it is organized in levels of index records. See also *alternate index.*

input/output control system (IOCS)

A group of IBM supplied routines that handle the transfer of data between main storage and auxiliary storage devices.

integrated communication adapter (ICA)

The part of a processor where multiple lines can be connected.

integrated console

In z/VSE, the service processor console available on IBM Z that operates as the z/VSE system console. The integrated console is typically used during IPL and for recovery purposes when no other console is available.

Interactive Computing and Control Facility (ICCF)

An IBM licensed program that serves as interface, on a time-slice basis, to authorized users of terminals that are linked to the system's processor.

interactive partition

An area of virtual storage for the purpose of processing a job that was submitted interactively via VSE/ICCF.

Interactive User Communication Vehicle (IUCV)

Programming support available in a VSE supervisor for operation under z/VM. The support allows users to communicate with other users or with CP in the same way they would with a non-preferred guest.

intermediate storage

Any storage device that is used to hold data temporarily before it is processed.

IOCS

Input/output control system.

IPL

Initial program load.

irrecoverable error

An error for which recovery is impossible without the use of recovery techniques external to the computer program or run.

IUCV

Interactive User Communication Vehicle.

J

JAR

Is a platform-independent file format that aggregates many files into one. Multiple applets and their requisite components (.class files, images, and sounds) can be bundled in a JAR file, and then downloaded to a web browser using a single HTTP transaction (much improving the download speed). The JAR format also supports compression, which reduces the files size (and further improves the download speed). The compression algorithm that is used is fully compatible with the ZIP algorithm. The owner of an applet can also digitally sign individual entries in a JAR file to authenticate their origin.

Java application

A Java program that runs inside the JVM of your web browser. The program's code resides on a local hard disk or on the LAN. Java applications might be large programs using graphical interfaces. Java applications have unlimited access to all your local resources.

Java bytecode

Bytecode is created when a file containing Java source language statements is compiled. The compiled Java code or "bytecode" is similar to any program module or file that is ready to be executed (run on a computer so that instructions are performed one at a time). However, the instructions in the bytecode are really instructions to the *Java Virtual Machine*. Instead of being interpreted one instruction at a time, bytecode is instead recompiled for each operating-system platform using a just-in-time (JIT) compiler. Usually, this enables the Java program to run faster. Bytecode is contained in binary files that have the suffix**.CLASS**

Java servlet

See *servlet.*

JHR

Job header record.

job accounting interface

A function that accumulates accounting information for each job step, to be used for charging the users of the system, for planning new applications, and for supervising system operation more efficiently.

job accounting table

An area in the supervisor where accounting information is accumulated for the user.

job catalog

A catalog made available for a job by means of the file name IJSYSUC in the respective DLBL statement.

job entry control language (JECL)

A control language that allows the programmer to specify how VSE/POWER should handle a job.

job step

In 1 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 trailer record (JTR)

As VSE/POWER parameter JTR, alias NJT. An NJE control record terminating a job entry in the input or output queue and providing accounting information.

K

key

In VSE/VSAM, one or several characters that are taken from a certain field (key field) in data records for identification and sequence of index entries or of the records themselves.

key sequence

The collating sequence either of records themselves or of their keys in the index or both. The key sequence is alphanumeric.

key-sequenced file

A VSE/VSAM file whose records are loaded in key sequence and controlled by an index. Records are retrieved and stored by keyed access or by addressed access, and new records are inserted in the file in key sequence.

KSDS

Key-sequenced data sets. See *key-sequenced file.*

L

label

1. An identification record for a tape, disk, or diskette volume or for a file on such a volume.
2. In assembly language programming, a named instruction that is generally used for branching.

label information area

An area on a disk to store label information that is read from job control statements or commands. Synonymous with *label area*.

Language Environment for z/VSE

An IBM software product that is the implementation of Language Environment on the VSE platform.

language translator

A general term for any assembler, compiler, or other routine that accepts statements in one language and produces equivalent statements in another language.

Large DASD

A DASD device that

1. Has a capacity exceeding 64 K tracks and
2. Does not have VSAM space created prior to VSE/ESA 2.6 that is owned by a catalog.

LE/VSE

Short form of Language Environment for z/VSE.

librarian

The set of programs that maintains, services, and organizes the system and private libraries.

library block

A block of data that is stored in a sublibrary.

library directory

The index that enables the system to locate a certain sublibrary of the accessed library.

library member

The smallest unit of a data that can be stored in and retrieved from a sublibrary.

line commands

In VSE/ICCF, special commands to change the declaration of individual lines on your screen. You can copy, move, or delete a line declaration, for example.

linkage editor

A program that is 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.

linkage stack

An area of protected storage that the system gives to a program to save status information for a branch and stack or a stacking program call.

link station

In SNA, the combination of hardware and software that allows a node to attach to and provide control for a link.

loader

A routine, commonly a computer program, that reads data or a program into processor storage. See also *relocating loader*.

local shared resources (LSR)

A VSE/VSAM option that is activated by three extra macros to share control blocks among files.

lock file

In a shared disk environment under VSE, a system file on disk that is used by the sharing systems to control their access to shared data.

logical partition

In LPAR mode, a subset of the server unit hardware that is defined to support the operation of a system control program.

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 that is used in programming to represent an I/O device address. *physical unit (PU), system services control point (SSCP), primary logical unit (PLU), and secondary logical unit (SLU).*
2. In SNA, a port through which a user accesses the SNA network,
   a. To communicate with another user and
   b. To access the functions of the SSCP. An LU can support at least two sessions. One with an SSCP and one with another LU and might be capable of supporting many sessions with other LUs.

logical unit name

In programming, a name that is used to represent the address of an input/output unit.

logical unit 6.2

A SNA/SDLC protocol for communication between programs in a distributed processing environment. LU 6.2 is characterized by

1. A peer relationship between session partners,
2. Efficient utilization of a session for multiple transactions,
3. Comprehensive end-to-end error processing, and
4. A generic Application Programming Interface (API) consisting of structured verbs that are mapped into a product implementation.

logons interpret interpret routine

In VTAM, an installation exit routine, which is associated with an interpret table entry, that translates logon information. It also verifies the logon.

LPAR mode

Logically partitioned mode. The CP mode that is available on the Configuration (CONFIG) frame when the PR/SM feature is installed. LPAR mode allows the operator to allocate the hardware resources of the processor unit among several logical partitions.

M

macro definition

A set of statements and instructions that defines the name of, format of, and conditions for generating a sequence of assembler statements and machine instructions from a single source statement.

macro expansion

See *macro generation*

macro generation

An assembler operation by which a macro instruction gets replaced in the program by the statements of its definition. It takes place before assembly. Synonymous with *macro expansion*.

macro (instruction)

1. In assembler programming, a user-invented assembler statement that causes the assembler to process a set of statements that are defined previously in the macro definition.

2. A sequence of VSE/ICCF commands that are defined to cause a sequence of certain actions to be performed in response to one request.

maintain system history program (MSHP)

A program that is used for automating and controlling various installation, tailoring, and service activities for a VSE system.

main task

The main program within a partition in a multiprogramming environment.

master console

In z/VSE, one or more consoles that receive all system messages, except for those that are directed to one particular console. Contrast this with the *user* console, which receives only those messages that are specifically directed to it, for example messages that are issued from a job that was submitted with the request to echo its messages to that console. The operator of a master console can reply to all outstanding messages and enter all system commands.

maximum (max) CA

A unit of allocation equivalent to the maximum control area size on a count-key-data or fixed-block device. On a CKD device, the max CA is equal to one cylinder.

memory object

Chunk of virtual storage that is allocated above the bar (2 GB) to be created with the IARV64 macro.

message

In VSE, a communication that is sent from a program to the operator or user. It can appear on a console, a display terminal or on a printout.

MSHP

See maintain system history program.

multitasking

Concurrent running of one main task and one or several subtasks in the same partition.

MVS

Multiple Virtual Storage. Implies MVS/390, MVS/XA, MVS/ESA, and the MVS element of the z/OS (OS/390) operating system.

N

NetView

A z/VSE optional program that is used to monitor a network, manage it, and diagnose its problems.

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 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 that is 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*.

Network Control Program (NCP)

An IBM licensed program that provides communication controller support for single-domain, multiple-domain, and interconnected network capability. Its full name is ACF/NCP.

network definition table (NDT)

In VSE/POWER networking, the table where every node in the network is listed.

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 several 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, a point in a network that is defined by a symbolic name. Synonymous with *network node*. See *major node and minor node*.

node type

In SNA, a designation of a node according to the protocols it supports and the network addressable units (NAUs) it can contain.

O

object module (program)

A program unit that is the output of an assembler or compiler and is input to a linkage editor.

online application program

An interactive program that is used at display stations. When active, it waits for data. Once input arrives, it processes it and send a response to the display station or to another device.

operator command

A statement to a control program, issued via a console or terminal. It causes the control program to provide requested information, alter normal operations, initiate new operations, or end existing operations.

optional licensed program

An IBM licensed program that a user can install on VSE by way of available installation-assist support.

output parameter text block (OPTB)

in VSE/POWER's spool-access support, information that is contained in an output queue record if a * $$ LST or * $$ PUN statement includes any user-defined keywords that have been defined for autostart.

P

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 fixing

Marking a page so that it is held in processor storage until explicitly released. Until then, it cannot be paged out.

page I/O

Page-in and page-out operations.

page pool

The set of page frames available for paging virtual-mode programs.

panel

The complete set of information that is shown in a single display on terminal screen. Scrolling back and forth through panels like turning manual pages. See also *selection panel*.

partition balancing

A z/VSE facility that allows the user to specify that two or more or all partitions of the system should receive about the same amount of time on the processor.

PASN-AL (primary address space number - access list)

The access list that is associated with a partition. A program uses the PASN-AL associated with its partition and the DU-AL associated with its task (work unit). See also *DU-AL*.

Each partition has its own unique PASN-AL. All programs running in this partition can access data spaces through the PASN-AL. Thus a program can create a data space, add an entry for it in the PASN-AL, and obtain the ALET that indexes the entry. By passing the ALET to other programs in the partition, the program can share the data space with other programs running in the same partition.

PDS

Page data sets.

phase

The smallest complete unit of executable code that can be loaded into virtual storage.

physical record

The amount of data that is transferred to or from auxiliary storage. Synonymous with *block*.

PNET

Programming support available with VSE/POWER; it provides for the transmission of selected jobs, operator commands, messages, and program output between the nodes of a network.

POWER

See *VSE/POWER*.

pregenerated operating system

An operating system such as z/VSE that is shipped by IBM mainly in object code. IBM defines such key characteristics as the size of the main control program, the organization, and size of libraries, and required system areas on disk. The customer does not have to generate an operating system.

preventive service

The installation of one or more PTFs on a VSE system to avoid the occurrence of anticipated problems.

primary address space

In z/VSE, the address space where a partition is executed. A program in primary mode fetches data from the primary address space.

primary library

A VSE library owned and directly accessible by a certain terminal user.

printer/keyboard mode

Refers to 1050 or 3215 console mode (device dependent).

Print Services Facility (PSF)/VSE

An access method that provides support for the advanced function printers.

private area

The virtual space between the shared area (24 bit) and shared area (31 bit), where (private) partitions are allocated. Its maximum size can be defined during IPL. See also *shared area*.

private memory object

Memory object (chunk of virtual storage) that is allocated above the 2 GB line (bar) only accessible by the partition that created it.

private partition

Any of the system's partitions that are not defined as shared. See also *shared partition*.

production library

1. In a pre-generated operating system (or product), the program library that contains the object code for this system (or product).
2. A library that contains data that is needed for normal processing. Contrast with *test library*.

programmer logical unit

A logical unit available primarily for user-written programs. See also *logical unit name*.

program temporary fix (PTF)

A solution or by-pass of one or more problems that are documented in APARs. PTFs are distributed to IBM customers for preventive service to a current release of a program.

PSF/VSE

Print Services Facility/VSE.

PTF

See *Program temporary fix*.

Q

Queue Control Area (QCA)

In VSE/POWER, an area of the data file, which might contain:

- Extended checkpoint information
- Control information for a shared environment.

queue file

A direct-access file that is maintained by VSE/POWER that holds control information for the spooling of job input and job output.

R

random processing

The treatment of data without respect to its location on disk storage, and in an arbitrary sequence that is governed by the input against which it is to be processed.

real address area

In z/VSE, processor storage to be accessed with dynamic address translation (DAT) off

real address space

The address space whose addresses map one-to-one to the addresses in processor storage.

real mode

In VSE, a processing mode in which a program might not be paged. Contrast with *virtual mode*.

recovery management support (RMS)

System routines that gather information about hardware failures and that initiate a retry of an operation that failed because of processor, I/O device, or channel errors.

refresh release

An upgraded VSE system with the latest level of maintenance for a release.

relative-record file

A VSE/VSAM file whose records are loaded into fixed-length slots and accessed by the relative-record numbers of these slots.

release upgrade

Use of the FSU functions to install a new release of z/VSE.

relocatable module

A library member of the type object. It consists of one or more control sections cataloged as one member.

relocating loader

A function that modifies addresses of a phase, if necessary, and loads the phase for running into the partition that is selected by the user.

remote interface

In the context of z/VSE, the remote interface allows a client to make method calls to an EJB although the EJB is on a remote z/VSE host. The container uses the remote interface to create client-side stubs and server-side proxy objects to handle incoming method calls from a client to an EJB.

remote procedure call (RPC)

1. A facility that a client uses to request the execution of a procedure call from a server. This facility includes a library of procedures and an external data representation.
2. A client request to service provider in another node.

residency mode (RMODE)

A program attribute that refers to the location where a program is expected to reside in virtual storage. RMODE 24 indicates that the program must reside in the 24-bit addressable area (below 16 megabytes), RMODE ANY indicates that the program can reside anywhere in 31-bit addressable storage (above or below 16 megabytes).

REXX/VSE

A general-purpose programming language, which is particularly suitable for command procedures, rapid batch program development, prototyping, and personal utilities.

RMS

Recovery management support.

RPG II

A commercially oriented programming language that is specifically designed for writing application programs that are intended for business data processing.

S

SAM ESDS file

A SAM file that is managed in VSE/VSAM space, so it can be accessed by both SAM and VSE/VSAM macros.

SCP

System control programming.

SDL

System directory list.

search chain

The order in which chained sublibraries are searched for the retrieval of a certain library member of a specified type.

second-level directory

A table in the SVA containing the highest phase names that are found on the directory tracks of the system sublibrary.

Secure Sockets Layer (SSL)

A security protocol that allows the client to authenticate the server and all data and requests to be encrypted. SSL was developed by Netscape Communications Corp. and RSA Data Security, Inc..

segmentation

In VSE/POWER, a facility that breaks list or punch output of a program into segments so that printing or punching can start before this program has finished generating such output.

selection panel

A displayed list of items from which a user can make a selection. Synonymous with *menu*.

sense

Determine, on request or automatically, the status or the characteristics of a certain I/O or communication device.

sequential access method (SAM)

A data access method that writes to and reads from an I/O device record after record (or block after block). On request, the support performs device control operations such as line spacing or page ejects on a printer or skip some tape marks on a tape drive.

service node

Within the VSE unattended node support, a processor that is 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 function in support of the system. See with *utility program*.

service refresh

A form of service containing the current version of all software. Also referred to as a *system refresh*.

service unit

One or more PTFs on disk or tape (cartridge).

shared area

In z/VSE, shared areas (24 bit) contain the Supervisor areas and SVA (24 bit) and shared areas (31 bit) the SVA (31 bit). Shared areas (24 bit) are at the beginning of the address space (below 16 MB), shared area (31 bit) at the end (below 2 GB).

shared disk option

An option that lets independent computer systems use common data on shared disk devices.

shared memory objects

Chunks of virtual storage allocated above the 2 GB line (bar), that can be shared among partitions.

shared partition

In z/VSE, a partition that is allocated for a program (VSE/POWER, for example) that provides services and communicates with programs in other partitions of the system's virtual address spaces. In most cases shared partitions are no longer required.

shared spooling

A function that permits the VSE/POWER account file, data file, and queue file to be shared among several computer systems with VSE/POWER.

shared virtual area (SVA)

In z/VSE, a high address area that contains a list system directory list (SDL) of frequently used phases, resident programs that are shared between partitions, and an area for system support.

SIT (System Initialization Table)

A table in CICS that contains data used the system initialization process. In particular, the SIT can identify (by suffix characters) the version of CICS system control programs and CICS tables that you have specified and that are to be loaded.

skeleton

A set of control statements, instructions, or both, that requires user-specific information to be inserted before it can be submitted for processing.

socksified

See *socks-enabled*.

Socks-enabled

Pertaining to TCP/IP software, or to a specific TCP/IP application, that understands the *socks protocol*. "Socksified" is a slang term for socks-enabled.

socks protocol

A protocol that enables an application in a secure network to communicate through a firewall via a *socks server*.

socks server

A circuit-level gateway that provides a secure one-way connection through a firewall to server applications in a nonsecure network.

source member

A library member containing source statements in any of the programming languages that are supported by VSE.

split

To double a specific unit of storage space (CI or CA) dynamically when the specified minimum of free space gets used up by new records.

spooling

The use of disk storage as buffer storage to reduce processing delays when transferring data between peripheral equipment and the processor of a computer. In z/VSE, this is done under the control of VSE/POWER.

Spool Access Protection

An optional feature of VSE/POWER that restricts individual spool file entry access to user IDs that have been authenticated by having performed a security logon.

spool file

1. A file that contains output data that is saved for later processing.
2. One of three VSE/POWER files on disk: queue file, data file, and account file.

SSL

See Secure Sockets Layer.

stacked tape

An IBM supplied product-shipment tape containing the code of several licensed programs.

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.

stand-alone program

A program that runs independently of (not controlled by) the VSE system.

startup

The process of performing IPL of the operating system and of getting all subsystems and applications programs ready for operation.

start option

In VTAM, a user-specified or IBM specified option that determines conditions for the time a VTAM system is operating. Start options can be predefined or specified when VTAM is started.

static partition

A partition, which is defined at IPL time and occupying a defined amount of virtual storage that remains constant. See also *dynamic partition*.

storage director

An independent component of a storage control unit; it performs all of the functions of a storage control unit and thus provides one access path to the disk devices that are attached to it. A storage control unit has two storage directors.

storage fragmentation

Inability to allocate unused sections (fragments) of storage in the real or virtual address range of virtual storage.

suballocated file

A VSE/VSAM file that occupies a portion of an already defined data space. The data space might contain other files. See also *unique file*.

sublibrary

In VSE, 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.

SVA

See shared virtual area.

Synchronous DataLink Control (SDLC)

A discipline for managing synchronous, code-transparent, serial-by-bit information transfer over a link connection. Transmission exchanges might be duplex or half-duplex over switched or non-switched links. The configuration of the link connection might be point-to-point, multipoint, or loop.

SYSRES

See system residence volume.

system control programming (SCP)

IBM supplied, non-licensed program fundamental to the operation of a system or to its service or both.

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

In z/VSE, a file that is used by the operating system, for example, the hardcopy file, the recorder file, the page data set.

System Initialization Table (SIT)

A table in CICS that contains data that is used by the system initialization process. In particular, the SIT can identify (by suffix characters) the version of CICS system control programs and CICS tables that you have specified and that are to be loaded.

system recorder file

The file that is used to record hardware reliability data. Synonymous with *recorder file*.

system refresh

See *service refresh*.

system refresh release

See *refresh release*.

system residence file (SYSRES)

The z/VSE system sublibrary IJSYSRS.SYSLIB that contains the operating system. It is stored on the system residence volume DORSES.

system residence volume (SYSRES)

The disk volume on which the system sublibrary is stored and from which the hardware retrieves the initial program load routine for system startup.

system sublibrary

The sublibrary that contains the operating system. It is stored on the system residence volume (SYSRES).

T

task management

The functions of a control program that control the use, by tasks, of the processor and other resources (except for input/output devices).

time event scheduling support

In VSE/POWER, the time event scheduling support offers the possibility to schedule jobs for processing in a partition at a predefined time once repetitively. The time event scheduling operands of the * $$ JOB statement are used to specify the wanted scheduling time.

TLS

See Transport Layer Security.

track group

In VSE/POWER, the basic organizational unit of a file for CKD devices.

track hold

A function that protects a track that is being updated by one program from being accessed by another program.

transaction

1. In a batch or remote batch entry, a job or job step. 2. In CICS TS, 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.
2. A given task can relate only to one operator.

transient area

An area within the control program that is used to provide high-priority system services on demand.

Transport Layer Security

The newest SSL cryptographic protocol. It provides additional strength to privacy and data integrity.

Turbo Dispatcher

A facility of z/VSE that allows to use multiprocessor systems (also called CEC: Central Electronic Complexes). Each CPU within such a CEC has accesses to be shared virtual areas of z/VSE: supervisor, shared areas (24 bit), and shared areas (31 bit). The CPUs have equal rights, which means that any CPU might receive interrupts and work units are not dedicated to any specific CPU.

U

UCB

Universal character set buffer.

universal character set buffer (UCB)

A buffer to hold UCS information.

UCS

Universal character set.

user console

In z/VSE, a console that receives only those system messages that are specifically directed to it. These are, for example, messages that are issued from a job that was submitted with the request to echo its messages to that console. Contrast with *master console*.

user exit

A programming service that is provided by an IBM software product that can be requested during the execution of an application program for the service of transferring control back to the application program upon the later occurrence of a user-specified event.

V

variable-length relative-record data set (VRDS)

A relative-record data set with variable-length records. See also *relative-record data set*.

variable-length relative-record file

A VSE/VSAM relative-record file with variable-length records. See also *relative-record file*.

VIO

See virtual I/O area.

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 addressability extension (VAE)

A storage management support that allows to use multiple virtual address spaces.

virtual address space

A subdivision of the virtual address area (virtual storage) available to the user for the allocation of private, nonshared partitions.

virtual disk

A range of up to 2 gigabytes of contiguous virtual storage addresses that a program can use as workspace. Although the virtual disk exists in storage, it appears as a real FBA disk device to the user program. All I/O operations that are directed to a virtual disk are intercepted and the data to be written to, or read from, the disk is moved to or from a data space.

Like a data space, a virtual disk can hold only user data; it does not contain shared areas, system data, or programs. Unlike an address space or a data space, data is not directly addressable on a virtual disk. To manipulate data on a virtual disk, the program must perform I/O operations.

Starting with z/VSE 5.2, a virtual disk may be defined in a shared memory object.

virtual I/O area (VIO)

An extension of the page data set; used by the system as intermediate storage, primarily for control data.

virtual mode

The operating mode of a program, where the virtual storage of the program can be paged, if not enough processor (real) storage is available to back the virtual storage.

virtual partition

In VSE, a division of the dynamic area of virtual storage.

virtual storage

Addressable space image for the user from which instructions and data are mapped into processor storage locations.

virtual tape

In z/VSE, a virtual tape is a file (or data set) 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 local system.
- 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.

volume ID

The volume serial number, which is a number in a volume label that is assigned when a volume is prepared for use by the system.

VRDS

Variable-length relative-record data sets. See *variable-length relative record file*.

VSAM

See *VSE/VSAM*.

VSE (Virtual Storage Extended)

A system that consists of a basic operating system and any IBM supplied and user-written programs that are required to meet the data processing needs of a user. VSE and hardware it controls form a complete computing system. Its current version is called z/VSE.

VSE/Advanced Functions

A program that provides basic system control and includes the supervisor and system programs such as the Librarian and the Linkage Editor.

VSE Connector Server

Is the host part of the VSE JavaBeans, and is started using the job STARTVCS, which is placed in the reader queue during installation of z/VSE. Runs by default in dynamic class R.

VSE/DITTO (VSE/Data Interfile Transfer, Testing, and Operations Utility)

An IBM licensed program that provides file-to-file services for disk, tape, and card devices.

VSE/ESA (Virtual Storage Extended/Enterprise Systems Architecture)

The predecessor system of z/VSE.

VSE/Fast Copy

A utility program for fast copy data operations from disk to disk and dump/restore operations via an intermediate dump file on magnetic tape or disk.

VSE/FCOPY (VSE/Fast Copy Data Set program)

An IBM licensed program for fast copy data operations from disk to disk and dump/restore operations via an intermediate dump file on magnetic tape or disk. There is also a stand-alone version: the FASTCOPY utility.

VSE/ICCF (VSE/Interactive Computing and Control Facility)

An IBM licensed program that serves as interface, on a time-slice basis, to authorized users of terminals that are linked to the system's processor.

VSE/ICCF library

A file that is composed of smaller files (libraries) including system and user data, which can be accessed under the control of VSE/ICCF.

VSE JavaBeans

Are JavaBeans that allow access to all VSE-based file systems (VSE/VSAM, Librarian, and VSE/ICCF), submit jobs, and access the z/VSE operator console. The class library is contained in the *VSEConnector.jar* archive. See also *JavaBeans*.

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 licensed program that is 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.

VSE/VSAM managed space

A user-defined space on disk that is placed under the control of VSE/VSAM.

W

wait for run subqueue

In VSE/POWER, a subqueue of the reader queue with dispatchable jobs ordered in execution start time sequence.

wait state

The condition of a processor when all operations are suspended. System recovery from a hard wait is impossible without performing a new system startup. See *hard wait*.

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.

work file

A file that is used for temporary storage of data being processed.

Numerics

24-bit addressing

Provides addressability for address spaces up to 16 megabytes.

31-bit addressing

Provides addressability for address spaces up to 2 gigabytes.

64-bit addressing

Provides addressability for address spaces up to 2 gigabytes and above.

Index

Product Number: 5686-CF8

SC33-8329-01