Interactive System Productivity Facility/ Program Development Facility Version 2 Release 2

Services

MVS

Publication Number
SC34-4023-1

File Number
S370/4300-39

Program Number
5665-317

Second Edition (September 1985)

This edition applies to Version 2, Release 2, of the program product Interactive System Productivity Facility/Program Development Facility (5665-317), and to all subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters. It is for use with the Interactive System Productivity Facility Version 2, Release 2 (5665-319), and OS/VS2 (MVS) Release 3.8 or later. In the Extended Architecture (XA) environment, it is for use with MVS/SP Version 2.

Changes are made periodically to the information herein. Therefore, before using this publication, consult the latest IBM System/370 and 4300 Processors Bibliography (GC20-0001) for the editions that are applicable and current.

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 program product in this document is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the address given below. Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Dept. T46, P. O. Box 60000, Cary, NC 27511. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

PREFACE

The Interactive System Productivity Facility (ISPF) and the ISPF/Program Development Facility (ISPF/PDF or PDF) are related IBM program products. Together, they are designed to improve user productivity in the development of applications, and contain special functions for the development, test, and use of interactive applications, called dialogs. Specifically:

- ISPF is a dialog manager for interactive applications. It provides control and services to permit execution of the dialogs.

- PDF is a dialog that aids in the development and testing of dialogs and other types of applications. It makes use of display terminals and an interactive environment to assist with many of a programmer's tasks.

This manual describes the library access services and the BROWSE and EDIT services of ISPF/PDF. It assumes that you are an application or systems programmer engaged in program development, and that you are familiar with the MVS environment.

The first chapter presents the services overview of the support provided by Version 2 of ISPF/PDF, and supplies general usage information. Chapter 2 describes the library management service that is required to initialize all the other services. Chapter 3 describes the BROWSE, EDIT, and EDREC services. Chapters 4 through 8 describe the remaining library access services, grouped according to logical function. Appendix A contains an alphabetized list of the PDF services, while Appendix B contains examples of using the services.

The following publications provide further information about ISPF and PDF:

ISPF/PDF for MVS Program Reference (SC34-4024) - Provides information on the use of PDF under MVS.

ISPF for MVS Dialog Management Services (SC34-4021) - Provides a detailed description of the dialog management services and related information required to develop an interactive application that runs under ISPF.

ISPF for MVS Dialog Management Services Examples (SC34-4022) - Provides a set of examples of the use of dialog management services.

ISPF and ISPF/PDF for MVS Installation and Customization (SC34-4019) - Provides information needed to install ISPF and ISPF/PDF and to custom tailor these products for a particular installation.

ISPF/PDF for MVS Library Management (SC34-4025) - Provides a detailed description on the use of the library management facility.

ISPF/PDF for MVS Edit Macros (SC34-4018) - Provides a detailed description on the use of the Edit macros.

In this document, the following notation conventions are used to describe PDF command formats:

- Uppercase commands and their operands must be entered as shown, but not necessarily in uppercase. Operands shown in lowercase are variables; you substitute your own values for them.

- Operands shown in brackets [] are optional, with a choice indicated by an OR symbol (|), or by stacking the operands. You may choose one or none; the defaults are underscored.

- Operands shown without brackets are required. If several operands are separated by an OR symbol (|), or stacked and shown in braces {}, you must select at least one of the choices.

CONTENTS

CHAPTER 1. INTRODUCTION

ISPF/PDF services provide the functions that are required to develop interactive MVS applications (ISPF dialogs). ISPF/PDF services consist of BROWSE and EDIT services and a set of library access services. The ISPF/PDF services will be used by the application programmers or systems programmers who are responsible for developing ISPF dialogs, and who are familiar with the MVS environment, the dialog management services, and PDF library management.

The library access services can be used to interact with the BROWSE and EDIT services. The BROWSE and EDIT services provide an interface to the Browse and Edit options of ISPF/PDF, and bypass the Browse and Edit entry panels.

The library access services provide an interface to ISPF libraries and to certain system data sets. (ISPF/PDF for MVS Program Reference contains an explanation of the ISPF library structure.)

The library access services may be used with the following types of data sets:

An ISPF library known by project, group, and type

A concatenated set of no more than four ISPF libraries on input

A single, existing MVS partitioned or sequential data set

The library access services only support data sets that reside on a single DASD volume with record format types F, FB, V, VB, or U. The data set organization must be either partitioned or sequential.

The ISPF/PDF services may be invoked by a program function dialog through a CALL to ISPEXEC or ISPLINK, or by a command function dialog (CLIST) through the ISPEXEC interface. For each service, the invocation will be shown through a call to ISPLINK and through the ISPEXEC interface.

The BROWSE, EDIT, and library access services are described in logical groups. Thus, services that deal with opening and closing data sets are discussed together, input and output functions are grouped, and so on. An alphabetical listing of all the services with a brief description of each, and references to where details can be found, appears in Appendix A.

The ISPF/PDF services allow users to access several PDF functions in a user-written dialog. A user can also access some MVS data sets from a user-written dialog using the library access services. In addition, the ISPF/PDF services can be used by the dialog writer to perform library management functions and to write Edit macros. For more information on library management and Edit macros, refer to ISPF/PDF for MVS Library Management and ISPF/PDF for MVS Edit Macros.

The ISPF/PDF services communicate with the user-written dialog by means of dialog variables. Thus, ISPF/PDF services can be used with dialog manager services. For information about dialog manager services and about writing dialogs, see ISPF/PDF for MVS Dialog Management Services.

INVOCATION OF SERVICES

This publication assumes that you have already invoked ISPF, established your startup data sets needed to invoke ISPF/PDF, and that you are familiar with ISPF dialog management services. These topics are discussed in ISPF for MVS Dialog Management Services.

Each ISPF/PDF service description shows the format for command invocation and for call invocation from a program module. The command invocation format is used from a CLIST or in option 7.6 of PDF (the dialog test facility).

Call invocation formats are shown in PL/I syntax. For example, ";" ends statements in the formats described. This is a PL/I convention, but syntax should be appropriate to the language being used.

Consider using ISPF/PDF model facilities (under Edit) when coding requests for ISPF services (see ISPF/PDF for MVS Program Reference for a description of these facilities).

Command Invocation

Library access services are invoked using the ISPEXEC command in a command procedure (CLIST) or while operating under option 7.6 of ISPF/PDF.

The general format for command invocation is:

```
ISPEXEC  service-name parameter1  parameter2  parameter3  ...
```

The "service-name" is alphabetic, up to eight characters long.

For some services, "parameter1" is a positional parameter and is required. Other parameters are keyword parameters. They may take either of two forms:

```
keyword
keyword(value)
```

Some keyword parameters are required and others are optional, as indicated for each service. Keyword parameters may be coded in any order. If duplicate keywords are coded, the last instance of the keyword is used.

CLIST variables consisting of a name preceded by an ampersand (&), may be used anywhere within the statement as the service name or as a parameter. Each variable is replaced with its current value prior to execution of the ISPEXEC command. (See OS/VS2 TSO Terminal User's Guide, GC28-0645, for a description of valid CLIST variable syntax.)

> **Note:** TSO CLIST attention exits are not recognized by ISPF and, if used, may cause unpredictable results.

Some ISPF/PDF services allow the names of dialog variables to be passed as parameters. Variable names are from one to eight alphanumeric characters. These names should not be preceded with an ampersand unless substitution is desired; for example:

```
ISPEXEC  LMINIT DATAID(XYZ) DDNAME(MYFILE)

SET &VNAME=XYZ
ISPEXEC  LMINIT DATAID(&VNAME) DDNAME(MYFILE)
```

In the first example, XYZ is the name of the dialog variable to be passed. In the second example, variable VNAME contains the name of the dialog variable to be passed.

Call Invocation

ISPF/PDF services are invoked from programs by invoking an ISPF subroutine interface. There are two basic call interfaces: ISPLINK and ISPEXEC. Alternate name forms, ISPLNK and ISPEX, are used in FORTRAN and Pascal programs because the length of a module name is limited to six characters in these languages. APL2 cannot be invoked with an ISPLINK call.

Other than for FORTRAN, Pascal, and APL2, the general call format for invoking ISPF/PDF services from functions using ISPLINK is:

```
CALL  ISPLINK (service-name, parameter1, parameter2, parameter3,... )
```

The parameters in this type of call statement are all positional; they must be coded in the order described for each service. Optional parameters may be omitted in a right-to-left dropout sequence. To obtain the default value for an optional parameter, code the parameter as one or more blanks (' '). Coding a blank parameter has the same effect as omitting it, and is used when parameters are to be dropped in other than a right-to-left sequence. When parameters are dropped in other than right-to-left sequence, the commas, followed by blanks, become required, as shown in the call format for each service.

The command function form for service requests (ISPEXEC) may be used in a program function by using the call format of ISPEXEC.

Other than for FORTRAN and Pascal, the general call format for invoking ISPF/PDF services from program functions using ISPEXEC is:

```
CALL  ISPEXEC (buf-len, buffer)
```

buf-len
Specifies a fullword fixed binary integer containing the length of buffer.

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure (CLIST).

All services that are valid through use of the ISPEXEC command procedure statements are valid through the CALL ISPEXEC interface.

The ISPEXEC call interface permits parameters to be specified as symbolic variables. Prior to a scan and syntax check of a statement, variable names (and the preceding ampersands) are replaced with the values of the corresponding variables. A single scan takes place. Maximum buffer size is 512K.

Standard register conventions are used. Registers 2-14 are preserved across the call.

> **Note:** The last parameter in the calling sequence must be indicated with a high-order "1" bit in the last entry of the address list. This high-order bit is automatically generated by PL/I, COBOL, and FORTRAN call statements. It requires use of the VL keyword in Assembler call statements.

FORTRAN or Pascal

The general call format for invoking ISPF/PDF services from FORTRAN or Pascal functions is either of the following:

```
lastrc = ISPLNK (service-name, parameter1, parameter2, parameter3,...)

lastrc = ISPEX (buf-len, buffer)
```

The parameters for ISPLNK and ISPEX are the same as those for ISPLINK and ISPEXEC.

lastrc is a FORTRAN integer variable in which the return code from the specified ISPF/PDF service is available. lastrc is any valid FORTRAN name.

For functions written in FORTRAN, arguments may be passed as FORTRAN variables or literals.

FORTRAN example:

```
  INTEGER      LASTRC(*4)
  CHARACTER    SERVIS(*8),DATAID(*8),OPTION(*8)
  DATA         SERVIS/'LMOPEN '/
  DATA         OPTION/'INPUT '/
                 .
                 .
                 .

  LASTRC=ISPLNK(SERVIS, DATAID, OPTION)
```

For FORTRAN service requests, parameter variables may be initialized using literals in assignment statements. Previously-defined constants must be used in assignment statements. For example:

```
  CHARACTER    LMOPEN(*8),SERVIS(*8)
  DATA         LMOPEN/'LMOPEN  '/
              .
              .
              .
  SERVIS=LMOPEN
```

APL2

For information about the use of APL2 with ISPF, refer to ISPF for MVS Dialog Management Services. The general format for invoking ISPF/PDF services from APL2 functions is:

```
lastrc <- ISPEXEC   character-vector
```

lastrc is an APL2 variable into which the return code of the service is to be stored. A character-vector is a single character vector that contains all parameters to be passed to the dialog service. The format is the same as dialog service statements for command languages. The first token in that vector must be the name of the service to be invoked.

Standard register conventions are used. Registers 2-14 are preserved across the call.

APL2 example:

```
/* OPEN THE DATA SET */
LASTCC <- ISPEXEC 'LMOPEN DATAID INPUT'
-> ( LASTCC = 0) / NORMALCONT
/* PROCESS ERRORS HERE */
    .
    .
    .
```

PL/I

In PL/I programs, the following declare statements should be included:

```
DECLARE ISPLINK   /*NAME OF ENTRY POINT*/
        ENTRY
        EXTERNAL  /*EXTERNAL ROUTINE*/
        OPTIONS(  /*NEEDED OPTIONS*/
        ASM,      /*DO NOT USE PL/I DOPE VECTORS*/
        INTER,    /*INTERRUPTS*/
        RETCODE); /*EXPECT A RETURN CODE*/
```

PL/I example:

```
DECLARE  SERVICE CHAR(8) INIT('LMOPEN  '),
         DATAID  CHAR(8),
         OPTION  CHAR(8) INIT('INPUT   ');
         . . .

CALL  ISPLINK (SERVICE, DATAID, OPTION);
```

For service calls in PL/I, parameter values may be initialized using literals in assignment statements, as in

```
SERVICE='LMOPEN';
```

COBOL

Some languages, such as COBOL, do not allow literals within a call statement. Use of literals is never required; all parameters may be specified as variables, as in the following examples:

COBOL example:

```
WORKING-STORAGE SECTION.
   77  SERVIS      PICTURE A(8) VALUE 'LMOPEN  '.
   77  DATAID      PICTURE A(8).
   77  OPTSHUN     PICTURE A(8) VALUE 'INPUT   '.
   . . .

PROCEDURE DIVISION.
   CALL 'ISPLINK' USING  SERVIS DATAID OPTSHUN.
```

For service calls in COBOL, parameter variables may be initialized using literals in assignment statements, as in

```
MOVE 'LMOPEN' TO SERVIS.
```

Parameters

The following types of parameters may appear in a calling sequence to ISPLINK or ISPLNK:

- Service name or keyword: A left-justified character string that must be coded as shown in the description of the particular service. The string may be up to eight characters long. It need not be delimited by a trailing blank.

- Single name: A left-justified character string. If the string is less than the maximum length for the particular parameter, it must have a trailing blank to delimit the end of the string. The minimum length for a single name is one character. The maximum length for most names is eight characters; the exceptions are data set name and volume serial.

- Numeric value: A fullword fixed binary number.

- Name list - string format: A list of dialog variable names coded as a character string. Each name is one to eight characters. The string must start with a left parenthesis and end with a right parenthesis. Within the parentheses, the names may be separated with commas or blanks; for example:

  ```
  '(AAA BBB CCC)' or '(AAA,BBB,CCC)'
  ```

  When the list consists of a single name, the parentheses are not required, but, if parentheses are not used, a trailing blank is required if the name is less than eight characters in length.

- Name list - structure format: A list of dialog variable names passed in a structure. Each name is one to eight characters. The structure must contain the following information in the following order:

  1. Count - Fullword fixed binary integer containing the number of names in the list.

  2. Reserved - Fullword fixed binary integer that must contain a value of either zero or eight.

  3. List of names - Each element in the list must be an 8-byte character string. Within each element, the name of the variable must be left-justified with trailing blanks.

- Subfield with keyword: A left-justified character string that must be coded exactly as shown. If the subfield does not contain the maximum number of characters, trailing blanks filling out the field must be specified. For example, choosing the NO option from STATS(YES|NO) results in 'NO ' passed as a parameter. (See the LMMLIST service.)

RETURN CODES FROM SERVICES

Each service returns a numeric code, called a return code, indicating the results of the operation. Return codes are grouped into two general categories:

- Normal completion (code 0).

- Error condition (codes 4, 8, 10, 12, 14, 16, and 20) - indicates that the service did not complete, or only partially completed, operation due to errors.

Return codes and their meanings vary for each service and are listed with each service description.

For command invocation, the code is returned in the CLIST variable LASTCC. For call invocation, the code is returned in register 15 or, in FORTRAN and Pascal programs, in registers 15 and 0. In APL2, the return code is placed on the execution stack by the function ISPEXEC.

Programs coded in FORTRAN or Pascal may examine the return code by using an integer variable, such as lastrc in the following example:

```
  lastrc = ISPLNK (service name, parameter1, parameter2, parameter3 ...)
```

Programs coded in PL/I may examine the return code by using the PLIRETV built-in function.

The following declare statements are required:

```
     DECLARE ISPLINK EXTERNAL ENTRY OPTIONS(ASM INTER RETCODE);
     DECLARE PLIRETV BUILTIN;
```

Programs coded in COBOL may examine the return code by using the RETURN-CODE built-in variable.

The action taken in the case of errors (return code 12 or higher) depends upon the error mode setting (see the CONTROL service in ISPF for MVS Dialog Management Services). There are two error modes:

- CANCEL - displays and logs a message, then terminates the dialog and redisplays the primary option menu.

- RETURN - formats an error message (but does not display or log it), then returns to the function that invoked the service, passing back the designated return code.

The dialog may use the dialog manager CONTROL service to set the error mode with a default of CANCEL. In CANCEL mode, control is usually not returned to the function that invoked the service. Consequently, the function will not see a return code of 12 or higher, and may not include logic to process errors of this severity. However, ISPLINK will return a code of 20 to the dialog when the code is caused by an invalid ISPF environment. (In this situation, ISPF cannot display a panel to indicate the error.)

In RETURN mode, control is returned to the function that invoked the service. That function must then have logic to handle return codes of 12 or higher.

The RETURN mode applies only to the function that set it with the CONTROL service. If a lower-level function is invoked, it starts out in CANCEL mode. When a function returns to the higher-level function that invoked it, the mode in which the higher-level function was operating is resumed.

For ISPF/PDF library access services, a single return code may have multiple error conditions. If an error occurs, an error message is formatted prior to returning to the function. The message ID is contained in system variable ZERRMSG. The short and long message text (in which substitutable variables have been resolved) is contained in system variables ZERRSM and ZERRLM, respectively. If a corresponding help panel was specified in the message definition, the name of the help panel is contained in system variable ZERRHM. If a corresponding alarm was set in the message definition, the value 'YES' is contained in the system variable ZERRALRM. Otherwise, the value 'NO' is contained in ZERRALRM. All of these system variables are in the top variable pool (that is, the function pool if it exists; if it does not exist, in the shared variable pool).

The function may display the message, log the message, or both, if desired, simply by invoking the appropriate service with the message ID ISRZ002. Examples:

```
  ISPEXEC  DISPLAY  MSG(ISRZ002)
  ISPEXEC  LOG  MSG(ISRZ002)
```

The short and long message text and the name of the corresponding help panel and the alarm setting are provided for the dialog writer's use.

CHAPTER 2. INITIALIZING AND INFORMATIONAL SERVICES

Most of the library access services require a data-id to be associated with the data set used by the services. The BROWSE and EDIT services can also be invoked by using a data-id. The data-id is an ISPF-generated representation of a data set. The data set can be an ISPF library, a concatenation of ISPF libraries, an MVS data set, or a concatenation of MVS data sets. The service LMINIT generates a unique data-id that becomes an input parameter to most of the library access services and, optionally, to the BROWSE and EDIT services. Hence, LMINIT is required by most of the other services.

For each LMINIT invocation, there should be a matching LMFREE service invoked. LMFREE removes the association of the data-id with the specified data-id. LMFREE should be invoked as soon as the dialog has completed its use of the data set. If LMFREE is not invoked, ISPF releases the ISPF library or data set when ISPF terminates.

The service LMQUERY can be used to determine how LMINIT has set parameters.

The services LMINIT, LMQUERY, and LMFREE are described in detail in this chapter.

LMINIT

The LMINIT service allows the dialog to associate a data-id with a specified data set. The data set can be an ISPF library or a concatenation of ISPF libraries or an MVS partitioned or sequential data set. The data-id is generated by LMINIT and may be used to identify the data set for processing by other library access services or the ISPF/PDF BROWSE or EDIT service. If the specified data set exists but has not been allocated, the LMINIT service allocates the data set. If two or more existing ISPF libraries are specified, the LMINIT service concatenates the libraries.

(example on page 110)

The input to the LMINIT service defines the physical and logical characteristics of the data set. This simplifies the invocation of the other library access services by supplying the information needed to invoke the service for a given data set. For instance, the dialog supplies the information required by the input fields on the ISPF/PDF Browse data entry panel to LMINIT. Subsequent invocations of the BROWSE service with that data set are made much simpler by using the data-id generated by the LMINIT service. The data-id is removed by the LMFREE service (see page 22).

```
ISPEXEC LMINIT DATAID(data-id-var)

               {PROJECT(project) GROUP1(group1) TYPE(type)          }
                   [GROUP2(group2)] [GROUP3(group3)] [GROUP4(group4)]

               {DATASET(dsname)                                     }

               {DDNAME(ddname)                                      }

               [VOLUME(serial)] [PASSWORD(password)]

               [ENQ(SHR|EXCLU|SHRW|MOD)] [ORG(org-var)]


CALL ISPEXEC (buf-len,buffer);


CALL ISPLINK ('LMINIT', data-id-var

                 {,project,group1,type          }
                    [,group2] [,group3] [,group4]

                 {,dsname                       }

                 {,ddname                       }

                 ,[serial] ,[password]

                 ,['SHR  '|'EXCLU'|'SHRW '|'MOD  '] ,[org-var]);
```

The data set (ISPF library, or MVS partitioned or sequential data set) is specified as a 3-level qualified name, or a dsname, or a ddname. LMINIT first looks for a ddname. If it finds a ddname, it uses that name. If it does not find a ddname, it looks for a dsname. If it finds a dsname, it uses that name. If it does not find a dsname, it takes the 3-level qualified name. If there is no 3-level qualified name, LMINIT issues an error message.

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id-var

Specifies the name of the variable in which the data-id to be associated with the data set is to be stored. A unique data-id is always generated by the LMINIT service. The data-id length is eight characters. The data-id is an input parameter to most of the other library access services, and to the BROWSE and EDIT services, but is an output parameter from the LMINIT service. The maximum length of this parameter is eight characters.

To invoke the service, the data-id variable name and a data set must be specified. The data set may be specified using either the name of an ISPF library (as specified in the project, group, and type parameters), or a dsname or a ddname. The ISPF library name and the dsname and ddname parameters, used to specify the data set, are described below.

project

Indicates the highest level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. This parameter is required if neither the dsname nor the ddname parameter is specified. The maximum length of this parameter is eight characters.

group1

Indicates the second-level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. This parameter is required if neither the dsname nor the ddname parameter is specified. The maximum length of this parameter is eight characters.

type

Indicates the third-level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. This parameter is required if neither the dsname nor the ddname parameter is specified. The maximum length of this parameter is eight characters.

group2

Continues the second-level qualifier above. It is not required, but if present it represents an ISPF library in a concatenation sequence. The maximum length of this parameter is eight characters.

group3

Continues the second-level qualifier above. It is not required, but if present it represents an ISPF library in a concatenation sequence. The maximum length of this parameter is eight characters.

group4

Continues the second-level qualifier above. It is not required, but if present it represents an ISPF library in a concatenation sequence. The maximum length of this parameter is eight characters.

dsname

Specifies the name of an existing MVS partitioned or sequential data set. A member name cannot be included in the dsname of a partitioned data set. The maximum length of this parameter is 46 characters, with two characters for quotes and 44 characters for the data set name, if the name has no prefix. A fully qualified data set name may be specified, enclosed in apostrophes. If the apostrophes are omitted, the TSO user prefix is automatically attached to the data set name.

ddname

Specifies the data set definition name of a data set that is already allocated to the TSO user before invocation of the LMINIT service. This can be done using the TSO ALLOCATE command or MVS job control language. The data set must be either partitioned or sequential.

If the ddname is allocated to one or more partitioned data sets with members, the member name is ignored, and the data set is treated as one partitioned data set by the library access services and the BROWSE and EDIT services. LMINIT allows up to four concatenated data sets.

Sequential data sets must be allocated either OLD, SHR, NEW, or MOD. If the ddname is allocated NEW, the record format, data set organization, record length, and block size must be specified when the ddname is allocated. For a partitioned data set, the number of directory blocks must also be specified when the ddname is allocated. The maximum length of this parameter is eight characters.

serial

Specifies the serial number of the DASD volume on which the data set resides. This parameter is associated with the dsname parameter above, but is required only if the data set is not cataloged. The maximum length of this parameter is six characters.

password

Specifies the MVS password of the data set. This parameter is required only if the data is password-protected. If the password is invalid, it will be detected by the LMOPEN service (page 40). The maximum length of this parameter is eight characters. (The password is not specified for RACF or PCF protected data sets.)

SHR|EXCLU|SHRW|MOD

Specifies the requirements for enqueueing the data set so that the dialog may use the data in the desired manner. This parameter is ignored if the ddname parameter is specified.

SHR indicates that the existing data set may be shared, for example, when only reading from the data set. This option can be specified when using the INPUT option of the LMOPEN service. SHR is the default.

EXCLU indicates that exclusive use of the data set is required; for example, when modifying the data set. This option can be specified for either the INPUT or OUTPUT option of the LMOPEN service.

SHRW permits a 'shared write' for a data set. This option is used by ISPF/PDF Edit. It is used only for a partitioned data set. In this way, more than one user may read from the data set, but members may be rewritten when necessary through an ENQ or DEQ used by Edit. Edit can, in this case, have the data-id open for INPUT and OUTPUT at the same time. SHRW is also used with the INPUT and OUTPUT options of the LMOPEN service.

MOD indicates that more records are to be added to the end of a sequential data set. MOD is used with the OUTPUT option of the LMOPEN service.

org-var
Specifies the name of the variable into which the organization of the data set is stored. 'PO' is stored into the variable if the data set is partitioned, and 'PS' if it is physical sequential. The maximum length of this parameter is eight characters.

The following return codes are possible:

0 - Normal completion

8 - Data id is not created; the error condition is described in the ISPF system dialog variables

12 - Invalid parameter value

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue

Example 1:

Invoke the LMINIT service to associate a data-id with a data set concatenated from the following ISPF libraries:

```
ISPF.TESTLIB1.PLIOPT
ISPF.TESTLIB2.PLIOPT
ISPF.TESTLIB3.PLIOPT
ISPF.TESTLIB4.PLIOPT
```

Store the generated data-id in variable DDVAR.

The three invocations of the LMINIT service for this example are:

- Using ISPEXEC

```
ISPEXEC LMINIT DATAID(DDVAR) PROJECT(ISPF) GROUP1(TESTLIB1)  +
               GROUP2(TESTLIB2) GROUP3(TESTLIB3)             +
               GROUP4(TESTLIB4) TYPE(PLIOPT)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
     LMINIT DATAID(DDVAR) PROJECT(ISPF) GROUP1(TESTLIB1)
            GROUP2(TESTLIB2) GROUP3(TESTLIB3)
            GROUP4(TESTLIB4) TYPE(PLIOPT)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
     CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
DCL DDVAR CHAR (8);
CALL ISPLINK('VDEFINE','DDVAR   ',DDVAR,'CHAR',
              LENGTH(DDVAR));
CALL ISPLINK('LMINIT','DDVAR   ','ISPF    ',
             'TESTLIB1','TESTLIB2',
             'TESTLIB3','TESTLIB4','PLIOPT ');
```

Example 2:

Invoke the LMINIT service for a 2-level dsname, 'SMITH.CLIST', using dsname.

Two invocations of the LMINIT service for this example are:

- Using ISPEXEC

```
ISPEXEC LMINIT DATAID(DDVAR)                +
               DATASET('SMITH.CLIST')       +
               ENQ(SHR)
```

- Calling ISPLINK

```
CALL ISPLINK('LMINIT','DDVAR ',
             ' ',' ',' ',' ',' ',' ',
             '''SMITH.CLIST ''',' ',
             ' ',' ','SHR  ');
```

Example 3:

Invoke the LMINIT service for a new data set, using ddname.

Two invocations of the LMINIT service for this example are:

- Using ISPEXEC

```
      ATTRIB MYLIST BLKSIZE(800)     +
             LRECL(80) RECFM(F B)    +
             DSORG(PS)
      ALLOC DDNAME(MYDD) NEW         +
             SPACE(1,1) TRACKS KEEP  +
             USING(MYLIST)
      ISPEXEC LMINIT DATAID(DDVAR) DDNAME(MYDD)
```

- Calling ISPLINK

  For the second invocation, assume DDNAME(MYDD) has been allocated to the user using JCL. The invocation of LMINIT is:

```
      CALL ISPLINK ('LMINIT','DDVAR ',
                    ' ',' ',' ',' ',' ',' ',
                    ' ','MYDD ');
```

LMQUERY

The LMQUERY service provides the dialog with selected information about the data set associated with the given data-id. The service sets the contents of the variables named with the information being requested. Blanks are returned in a given variable if no value is applicable. Thus, LMQUERY returns only values specified by LMINIT. For example, if DATASET was not used in LMINIT, DATASET in LMQUERY would have blanks.

```
ISPEXEC LMQUERY DATAID(data-id)

         [PROJECT(proj-var)]  [GROUP1(group1-var)]

         [GROUP2(group2-var)]  [GROUP3(group3-var)]

         [GROUP4(group4-var)]  [TYPE(type-var)]

         [DATASET(dsn-var)]  [DDNAME(ddn-var)]

         [VOLUME(serial-var)]  [ENQ(enq-var)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK ('LMQUERY',data-id

             ,[proj-var] ,[group1-var]

             ,[group2-var] ,[group3-var]

             ,[group4-var] ,[type-var]

             ,[dsn-var] ,[ddn-var]

             ,[serial-var]

             ,[' ',' ',' ',' ',enq-var]);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer

Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id

Specifies the data-id associated with the data set about which information is being requested. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

Every other parameter of the service is a variable name. The information returned in each of these variables comes from the parameters of the LMINIT service (page 12). The enqueue requirement of the data set is returned to the enq-var parameter. Blanks are returned in any variable for which there is no applicable value.

The following return codes are possible:

0 - Normal completion

4 - No applicable information available for a specified keyword, blanks being returned

10 - No data set is associated with the given data-id (that is, LMINIT has not been completed)

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue

Example:

Invoke the LMQUERY service to provide information about a data set that is allocated using the LMINIT service with an ISPF library name. Use variable PRJV for the highest level qualifier (project), GRP1V, GRP2V, GRP3V, and GRP4V for the second level qualifiers (group), and TYPEV for the third level qualifier (type) of the libraries.

The three ways of invoking the LMQUERY service for this example are shown on the next page.

- Using ISPEXEC

```
     ISPEXEC LMQUERY DATAID(&DDVAR) PROJECT(PRJV) GROUP1(GRP1V) +
                     GROUP2(GRP2V) GROUP3(GRP3V) GROUP4(GRP4V)  +
                     TYPE(TYPEV)
```

- Calling ISPEXEC

  Set the program variable BUFFER to contain:

```
     LMQUERY DATAID(&DDVAR) PROJECT(PRJV) GROUP1(GRP1V)
             GROUP2(GRP2V) GROUP3(GRP3V) GROUP4(GRP4V)
             TYPE(TYPEV)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
     CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
     CALL ISPLINK('LMQUERY',DDVAR,'PRJV ','GRP1V ','GRP2V ',
                            'GRP3V ','GRP4V ','TYPEV ');
```

LMFREE

The LMFREE service removes the data-id and releases the ISPF library(s) or data set associated with the specified data-id. If the data set is still open, it is closed. If the data-id is not allocated using the DDNAME parameter in LMINIT, the allocation for the data set is also freed. In this case, if the data-id represents a concatenated set of ISPF libraries, the data sets are deconcatenated and freed. After LMFREE is invoked, that data-id can no longer be used to identify the data set for processing by other library access services.

An LMFREE invocation should be issued for each LMINIT invocation at the time the dialog finishes its use of the data set.

```
ISPEXEC LMFREE DATAID(data-id)


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMFREE',data-id);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id
Specifies the data-id associated with the data set to be released. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

The following return codes are possible:

0 - Normal completion

8 - Free data set failed; the error condition is described in the ISPF system dialog variables

10 - No data set is associated with the given data-id (that is, LMINIT has not been completed)

20 - Severe error, unable to continue

Example:

Invoke the LMFREE service to release the data set associated with the data-id in variable DDVAR.

The three ways of invoking the LMFREE service for this example are:

- Using ISPEXEC

  ```
  ISPEXEC LMFREE DATAID(&DDVAR)
  ```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

  ```
  LMFREE DATAID(&DDVAR)
  ```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

  ```
  CALL ISPEXEC (BUFLEN, BUFFER);
  ```

- Calling ISPLINK

  ```
  CALL ISPLINK('LMFREE',DDVAR);
  ```

CHAPTER 3. BROWSE AND EDIT SERVICES

The BROWSE and EDIT services provide an interface to the Browse and Edit options of ISPF/PDF, and bypass the browse and edit entry panels. These services can be used for any data set, ISPF library, or concatenation of ISPF libraries that can be allocated using the LMINIT service. The edit recovery service, EDREC, allows you to recover from a system failure or task abend which occurred during a previous invocation of Edit. A detailed description of each service, including syntax, return codes, and examples demonstrating how the service is invoked, is given in this chapter.

BROWSE

The BROWSE service provides an interface to the ISPF/PDF Browse option, and bypasses the display of the Browse entry panel. The BROWSE service permits the creation of a customized panel for displaying data. The service may be used recursively. BROWSE may be used to view any data set, ISPF library, or concatenation of ISPF libraries that can be allocated using the LMINIT service. It may not be used by a PL/I main program that also uses subtasking.

See ISPF/PDF for MVS Program Reference for a complete description of the Browse option.

```
ISPEXEC  BROWSE  DATASET(dsname)  [VOLUME(serial)]

                                  [PASSWORD(pswd-value)]

                                  [PANEL(panel-name)]

                                  [FORMAT(format-name)]

                                  [MIXED(YES|NO)]
OR

ISPEXEC  BROWSE  DATAID(data-id)  [MEMBER(member-name)]

                                  [PANEL(panel-name)]

                                  [FORMAT(format-name)]

                                  [MIXED(YES|NO)]

CALL ISPEXEC (buf-len, buffer);

CALL  ISPLINK ('BROWSE', {dsname} ,[serial]

                                  ,[pswd-value]

                                  ,[panel-name]

                                  {data-id}

                                  ,[member-name]

                                  ,[format-name]

                                  ,['YES'|'NO ']);
```

buf-len

Specifies a fullword fixed binary integer containing the length of "buffer."

buffer

Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

dsname

Specifies the name of the data set to be browsed. A fully qualified data set name may be specified, enclosed in apostrophes. If the apostrophes are omitted, the TSO user prefix is automatically attached to the data set name. For partitioned data sets, a member name may be specified, enclosed in parentheses. If a member name is not specified, a member selection list will be displayed. The maximum length of this parameter is 56 characters.

serial

Specifies the serial number of the volume on which the data set resides. If this parameter is omitted or coded as blank, the system catalog is searched for the data set name. The maximum length of this parameter is six characters.

pswd-value

Specifies the password if the data set has MVS password protection. (The password is not specified for RACF or PCF protected data sets.)

panel-name

Specifies the name of the panel to be used for displaying the data, which may be a customized panel that you provide. See ISPF and ISPF/PDF for MVS Installation and Customization for information on developing a customized panel.

data-id

Specifies the data-id that was returned from the LMINIT service. The maximum length of this parameter is eight characters.

> **Note:** The LMINIT service may be used in either of two ways before invoking the BROWSE service:

- LMINIT may be used to allocate existing data set(s) by specifying a data set name or ISPF library qualifiers. LMINIT returns a data-id as output. This data-id (rather than a data set name) is then passed as input to the BROWSE service.

- The dialog may allocate its own data set(s) using the TSO ALLOCATE COMMAND or MVS dynamic allocation, and then pass the ddname to LMINIT. Again, a data-id is returned as output from LMINIT and subsequently passed to the BROWSE service. This procedure is referred to as the "ddname interface" to BROWSE. It is particularly useful for browsing VIO data sets, which cannot be accessed by data set name because they are not cataloged.

member-name

Specifies a member of an ISPF library or an MVS partitioned data set. If a member name is not specified, a member selection list is displayed for the ISPF concatenation of libraries or ISPF partitioned data sets.

format-name

Specifies the name of the format to be used to reformat the data. The format-name parameter is provided for the support of the IBM 5550 terminal using the Double-Byte Character Set (DBCS).

YES|NO

If YES is specified, the BROWSE service treats the data as mixed mode DBCS data. If NO is specified, the data is treated as EBCDIC data.

Note: In TSO, a member name (enclosed in parentheses) may be included as part of the dsname specification when the DATASET keyword is used.

The dialog may allocate its own data set(s) using the TSO ALLOCATE COMMAND or MVS dynamic allocation, and then pass the ddname to LMINIT. A data-id is returned as output from LMINIT and subsequently passed to the BROWSE service. This procedure is referred to as the "ddname interface" to BROWSE. It is particularly useful for browsing VIO data sets, which cannot be accessed by data set name because they are not cataloged. For an example, see "Example 2: Using the ddname Interface to Browse and Edit" on page 106.

The following return codes are possible:

0 - Normal completion

12 - Zero-length data (empty sequential data set or zero length member of a partitioned data set)

14 - Specified member not found

16 - No members in library (when member list was requested)

20 - Severe error, unable to continue

Example:

Invoke the BROWSE service for TELOUT, a member of the ISPFPROJ.FTOUTPUT data set.

The three ways of invoking BROWSE for this example are:

- Using ISPEXEC

```
        ISPEXEC BROWSE DATASET('ISPFPROJ.FTOUTPUT(TELOUT)')
```

  or

```
        ISPEXEC LMINIT DATAID(DDBROW)    +
                       DATASET('ISPFPROJ.FTOUTPUT')
        ISPEXEC BROWSE DATAID(&DDBROW) MEMBER(TELOUT)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
        BROWSE DATASET('ISPFPROJ.FTOUTPUT(TELOUT)')
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
        CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
        CALL ISPLINK ('BROWSE','ISPFPROJ.FTOUTPUT(TELOUT)');
```

EDIT

The EDIT service provides an interface to the ISPF/PDF editor and bypasses the display of the Edit entry panel. The EDIT interface permits the creation of a customized panel for displaying data, as well as the specification of an initial macro and the edit profile to be used. EDIT may be used to view any data set, ISPF library, or concatenation of ISPF libraries that can be allocated using the LMINIT service. The service may be used recursively, either through nested dialogs, or by entering an EDIT command while editing. The EDIT service may not be issued by a PL/I main program that also uses subtasking. See ISPF/PDF for MVS Program Reference for a complete description of the editor.

When EDIT is operating in recovery mode, an audit trail of user interactions is automatically recorded in a PDF controlled data set. Following a system failure, you can use the audit trail to recover the data set you were editing.

Note: Dialogs that invoke the EDIT service should invoke the EDREC service first to initialize edit recovery, because the EDIT service does not perform edit recovery.

```
ISPEXEC  EDIT  DATASET(dsname)  [VOLUME(serial)]

                                [PASSWORD(pswd-value)]

                                [PANEL(panel-name)]

                                [MACRO(macro-name)]

                                [PROFILE(profile-name)]

                                [FORMAT(format-name)]

                                [YES|NO]

OR

ISPEXEC  EDIT  DATAID(data-id)  [MEMBER(member-name)]

                                [PANEL(panel-name)]

                                [MACRO(macro-name)]

                                [PROFILE(profile-name)]

                                [FORMAT(format-name)]

                                [YES|NO]

CALL ISPEXEC (buf-len, buffer);

CALL  ISPLINK ('EDIT', {dsname} ,[serial]

                                ,[pswd-value] ,[panel-name]

                                ,[macro-name] ,[profile-name]

                                {data-id}

                                ,[member-name]

                                ,[format-name]

                                ,['YES'|'NO ']);
```

buf-len

Specifies a fullword fixed binary integer containing the length of "buffer."

buffer

Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

dsname

Specifies, in TSO syntax, the name of the data set to be edited. This is equivalent to the "other" data set name on the Edit entry panel. A fully qualified data set name may be specified, enclosed in apostrophes. If the apostrophes are omitted, the TSO user prefix is automatically attached to the data set name. For partitioned data sets, a member name may be specified, enclosed in parentheses. If a member name is not specified, a member selection list is displayed. The maximum length of this parameter is 56 characters.

serial

Specifies the volume serial on which the data set resides. If this parameter is omitted or coded as blank, the system catalog is searched for the data set name. The maximum length of this parameter is six characters.

pswd-value

Specifies the password if the data set has MVS password protection. (The password is not specified for RACF or PCF protected data sets.)

member-name

Specifies the member of a library or an MVS partitioned data set. If a member name is not specified, a member selection list is displayed.

> **Note:** In TSO, a member name (enclosed in parentheses) may be included as part of the dsname specification when the DATASET keyword is used.

panel-name

Specifies the name of the panel to be used for displaying the data. This may be a customized panel provided by the dialog developer. See ISPF and ISPF/PDF for MVS Installation and Customization for information on developing a customized panel.

macro-name

Specifies the name of an initial macro to be executed after the data is read, but before it is displayed.

profile-name

Specifies the name of the edit profile to be used. If a profile name is not specified, the profile name defaults to the data "type" (ISPF library type, or last qualifier of the "other" TSO data set name).

data-id

Specifies the data-id that was returned from the LMINIT service.

format-name

Specifies the name of the format to be used to reformat the data. The format-name parameter is provided for the support of the IBM 5550 terminal using the Double-Byte Character Set (DBCS).

YES|NO

If YES is specified, the EDIT service treats the data as mixed mode DBCS data. If NO is specified, the data is treated as EBCDIC data.

Note: The LMINIT service may be used in either of two ways before invoking the EDIT service:

- LMINIT may be used to allocate existing data set(s) by specifying a data set name or ISPF library qualifiers. LMINIT returns a data-id as output. This same data-id (rather than a data set name) is then passed as input to the EDIT service.

- The dialog may allocate its own data set(s) using the TSO ALLOCATE COMMAND or MVS dynamic allocation, and then pass the ddname to LMINIT. Again, a data-id is returned as output from LMINIT and subsequently passed to the EDIT service. This procedure is referred to as the "ddname interface" to EDIT. It is particularly useful for editing VIO data sets, which cannot be accessed by data set name because they are not cataloged. For an example, see "Example 2: Using the ddname Interface to Browse and Edit" on page 106.

The following return codes are possible:

0 - Normal completion, data was saved

4 - Normal completion, data was not saved

16 - No members in library (when member list was requested)

20 - Severe error, unable to continue

<u>Example</u>:

Invoke the EDIT service for TELOUT, a member of the ISPFPROJ.FTOUTPUT data set.

The three ways of invoking EDIT for this example are:

- Using ISPEXEC

```
          ISPEXEC EDIT DATASET('ISPFPROJ.FTOUTPUT(TELOUT)')
    or
          ISPEXEC LMINIT DATAID(EDT) DATASET('ISPFPROJ.FTOUTPUT')
          ISPEXEC EDIT DATAID(&EDT) MEMBER(TELOUT)
```

- Calling ISPEXEC

  Set the program variable BUFFER to contain:
  ```
          EDIT DATASET(ISPFPROJ.FTOUTPUT(TELOUT))
  ```
  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:
  ```
          CALL ISPEXEC (BUFLEN, BUFFER);
  ```

- Calling ISPLINK

  ```
          CALL ISPLINK ('EDIT','ISPFPROJ.FTOUTPUT(TELOUT) ');
  ```

EDREC

The EDREC service initializes an edit recovery table, determines whether recovery is pending, and takes the action specified by the first argument.

> **Note:** Dialogs that invoke the EDIT service should invoke the EDREC service first to initialize edit recovery, because the EDIT service does not perform edit recovery.

When you invoke the EDREC service, EDREC displays a special panel. You can recover data, cancel recovery, defer recovery until a later time, or enter the END command to return to the next sequential command in your command invocation or to the next sequential instruction in your program.

```
ISPEXEC  EDREC {INIT    [CMD(command-name)]}

               {QUERY   }

               {PROCESS [PASSWORD(pswd-value)] [DATAID(data-id)]}

               {CANCEL }

               {DEFER   }


CALL ISPEXEC (buf-len, buffer);

CALL ISPLINK ('EDREC' {,'INIT    ' [,command-name]}

                      {,'QUERY   '}

                      {,'PROCESS' [,pswd-value] [,data-id]}

                      {,'CANCEL '}

                      {,'DEFER   '});
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

INIT

Specifies that an edit recovery table should be initialized in the user's profile library if one does not already exist for the current application.

command-name

Specifies a command (CLIST-EXEC) that is to be invoked to initialize the table. If this parameter is omitted, the INIT option invokes a PDF-supplied command (named ISREDRTI) that creates an 8-row edit recovery table (permitting eight levels of concurrent edit sessions with recovery active. The edit sessions may be due to recursion or split screen usage.)

If an application-supplied command is specified with the INIT option, the command should be patterned after ISREDRTI. It may create a different number of rows, use a different naming convention for the backup data sets, or specify "keep" (instead of "delete") as the backup data set disposition. The format of the edit recovery table must be the same as specified in ISREDRTI.

QUERY

Specifies that the edit recovery table should be searched for a pending recovery. When the QUERY option is specified, EDREC scans the edit recovery table for an entry containing a recovery pending condition. If an entry is found (return code 4), the dialog must then call EDREC with the PROCESS, CANCEL, or DEFER option to dispose of the pending recovery condition.

EDREC QUERY is normally used in a loop, since there may be more than one pending recovery (due to recursion or split screen usage of the dialog). Each subsequent call to EDREC with the QUERY option scans the table starting at the entry after the last one that was found. A typical loop (in pseudo-code) is as follows:

```
SET DONE = NO
DO UNTIL DONE = YES
  ISPEXEC EDREC QUERY
  IF RETCODE = 4 THEN
    ISPEXEC EDREC PROCESS
  ELSE
    SET DONE = YES
  ENDIF
ENDDO
```

Note: As shown in the above example, EDREC QUERY must be used before each invocation of any of the following EDREC functions: PROCESS, CANCEL, DEFER.

The following variables are stored in the dialog function pool when EDREC is called with the QUERY option and recovery is pending (return code 4):

ZEDBDSN - Backup data set name

ZEDTDSN - Target data set name

ZEDTMEM - Target member name (if applicable)

ZEDTVOL - Volume serial of target data set (if a volume serial was specified on invocation of the EDIT service)

ZEDROW - Row number of entry in edit recovery table

These variables may be interrogated by the dialog and may be used to display information to the end user. They must not be modified by the dialog. If EDREC QUERY indicates that recovery is not pending, the above variables are not meaningful.

PROCESS

Specifies that edit recovery should proceed.

pswd-value

Specifies the MVS password of the target data set. This parameter is valid only with the PROCESS option.

data-id

Specifies where the recovered data will be saved. If this parameter is omitted, EDREC attempts to save the recovered data in the same data set that was being edited when the system failure occurred.

> **Note:** Before using the data-id parameter, the dialog must first invoke the LMINIT service to specify the target data set (by data set name or ddname) and then pass the data-id to the EDREC service. (This procedure may also be used to control the allocation of the target data set for recovery, even if it is not the original data set being edited.) This procedure must be used if the data set being edited was originally specified to the EDIT service by the ddname interface.

CANCEL

Specifies cancellation of edit recovery. The backup data set is scratched and the corresponding entry in the edit recovery table is freed.

DEFER

Specifies that edit recovery is to be deferred. Recovery is cancelled, but the backup data set is saved so that recovery can be processed at another time.

The following return codes are possible:

```
 0 - Normal return
     INIT    - edit recovery table was successfully created
     QUERY   - recovery is not pending
     PROCESS - recovery was completed, and the data was saved

 4 - Normal return
     INIT    - edit recovery table already exists for current
               application
     QUERY   - entry found in edit recovery table (recovery
               is pending)
     PROCESS - recovery was completed but user did not save data

20 - Severe error, unable to continue
```

Example:

Invoke the EDREC service for INIT, then for QUERY, then for PROCESS, and finally for CANCEL.

```
ISPEXEC EDREC INIT
    /* This will create an edit */
    /* recovery table if one does */
    /* not exist. */

ISPEXEC EDREC QUERY
    /* This will scan the edit */
    /* recovery table for a recovery */
    /* pending condition. */

ISPEXEC EDREC PROCESS
    /* This recovers the backup file */
    /* and displays the recovered */
    /* data for editing. */

ISPEXEC EDREC CANCEL
    /* This erases the backup data */
    /* set and frees the corresponding */
    /* entry in the edit recovery table. */
```

CHAPTER 4. OPEN AND CLOSE SERVICES

If the data contained in a data set associated with a given data-id is to be useful in a dialog, the data set must be prepared for processing. That is, the data set must be "opened." After the data set has been opened, data can be read from the data set or written to the data set. LMOPEN is the library access service that opens the data set or ISPF library associated with a given data-id.

When the processing of data from an opened data set is complete, that data set should be "closed." That is, the data set is made inaccessible for processing until the next time it is opened. LMCLOSE is the library access service that closes the data set associated with a given data-id.

LMOPEN

The LMOPEN service opens the data set associated with a given data-id. The LMINIT service must be completed before LMOPEN can be used.

```
ISPEXEC LMOPEN DATAID(data-id)

               [OPTION(INPUT|OUTPUT)]

               [LRECL(lrecl-var)]

               [RECFM(recfm-var)]

               [ORG(org-var)]

CALL ISPEXEC (buf-len, buffer);

CALL ISPLINK ('LMOPEN',data-id

                      ,['INPUT '|'OUTPUT']

                      ,[lrecl-var]

                      ,[recfm-var]

                      ,[org-var]);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id
Specifies the data-id associated with the data set to be opened. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

INPUT |OUTPUT
Specifies whether the data set is to be opened for reading or writing. The default is INPUT.

INPUT (the default) specifies that the dialog invoking the service will be reading from the data set, using the LMMFIND and LMGET services. The allocation enqueue requirement can be SHR, EXCLU, or SHRW.

OUTPUT specifies that the dialog invoking the service will be writing to the data set, using the LMPUT and LMMADD or LMMREP services, or changing the data set, using the LMMDEL or LMMREN services. The allocation enqueue requirement can be EXCLU, SHRW, or MOD. If the data set is allocated SHRW, ISPF uses the RESERVE macro to reserve the DASD volume to the user when the data set is opened for output. This DASD volume remains reserved to the user until the LMCLOSE service is performed. The data-id must represent a single data set, not a concatenation of data sets.

lrecl-var
Specifies the name of the variable into which the actual data record length (or,if the record format is of variable length, the maximum data record length) is to be stored. This is an output parameter. The maximum length of this parameter is eight characters.

recfm-var
Specifies the name of the variable into which the record format code is to be stored. This is an output parameter. This variable must contain at least four characters. The maximum length of this parameter is eight characters.

org-var
Specifies the name of the variable into which the organization of the data set is stored. This is an output parameter. 'PO' is stored in the variable if the data set is partitioned, and 'PS' if it is physical sequential. The maximum length of this parameter is eight characters.

The following return codes are possible:

0 - Normal completion

8 - Data set could not be opened

10 - No data set associated with the given data-id (that is, LMINIT has not been completed)

12 - Invalid parameter value
- Data set is already open
- Cannot open concatenated data sets for output
- Cannot open a data set allocated SHR for output
- Cannot open a data set allocated MOD for input

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue

Example:

Invoke the LMOPEN service to open the data set associated with the data-id in variable DDVAR for reading. The record length is to be returned in variable DLVAR, the record format in RFVAR, and the data set organization in ORGVAR.

The three ways of invoking the LMOPEN service for this example are:

- Using ISPEXEC

```
        ISPEXEC LMOPEN DATAID(&DDVAR) OPTION(INPUT)      +
                       LRECL(DLVAR)   RECFM(RFVAR)       +
                       ORG(ORGVAR)
```

- Calling ISPEXEC

  Set the program variable BUFFER to contain:

```
        LMOPEN DATAID(&DDVAR) OPTION(INPUT)
               LRECL(DLVAR)   RECFM(RFVAR)
               ORG(ORGVAR)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
        CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
        CALL ISPLINK('LMOPEN',DDVAR,'INPUT ',
                              'DLVAR ','RFVAR ',
                              'ORGVAR ');
```

LMCLOSE

The LMCLOSE service closes the data set associated with a given data-id. A matching LMCLOSE invocation should be issued for every invocation of LMOPEN to free the data set as soon as possible. In the case where LMINIT is issued with an enqueue requirement of SHRW and LMOPEN is issued with the OUTPUT option, it is essential that an LMCLOSE is issued as soon as the dialog has finished processing the data set, since the DASD volume is reserved until LMCLOSE is invoked. On output, if the data set is sequential, the last physical block is written by the LMCLOSE service.

```
ISPEXEC LMCLOSE DATAID(data-id)
```

```
CALL ISPEXEC (buf-len, buffer);
```

```
CALL ISPLINK ('LMCLOSE',data-id);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id
Specifies the data-id associated with the data set to be closed. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

The following return codes are possible:

0 - Normal completion

8 - Data set is not open

10 - No ISPF library or data set is associated with the given data-id (that is, LMINIT has not been completed)

20 - Severe error, unable to continue

<u>Example</u>: Invoke the LMCLOSE service to close the data set associated with the data-id in variable DDVAR.

The three ways of invoking the LMCLOSE service for this example are:

- Using ISPEXEC

  ```
          ISPEXEC LMCLOSE DATAID(&DDVAR)
  ```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

  ```
          LMCLOSE DATAID(&DDVAR)
  ```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

  ```
          CALL ISPEXEC (BUFLEN, BUFFER);
  ```

- Calling ISPLINK

  ```
          CALL ISPLINK('LMCLOSE',DDVAR);
  ```

CHAPTER 5. INPUT/OUTPUT SERVICES

Once a data set has been opened, it can be processed. Two important operations that can be performed on a data set are reading records from the data set and writing records to the data set.

The LMGET service reads one logical record from the data set associated with the given data-id.

The LMPUT service writes one logical record to the data set associated with the given data-id.

LMGET

The LMGET service reads one logical record from the sequential data set or member of the partitioned data set associated with the given data-id. Completion of the LMINIT and LMOPEN services for the data set is required before LMGET is invoked.

If the data set to be processed is a sequential data set, the first LMGET to read from it will begin at the first logical record. Subsequent invocation will read succeeding logical records; that is, the second invocation reads the second logical record, the third invocation reads the third logical record, and so on.

If the data set is an ISPF library or MVS partitioned data set, previous completion of the LMMFIND service is required in addition to completion of LMINIT and LMOPEN. In this case, the LMGET service will read from the last member referenced by the LMMFIND service in the data set being processed. Thus, if LMMFIND is issued referencing member A, LMGET will read from member A. If another LMMFIND is issued referencing member B, LMGET will read from member B, not member A.

```
ISPEXEC LMGET DATAID(data-id)

              MODE(MOVE|LOCATE|INVAR)

              DATALOC(dataloc-var)

              DATALEN(datalen-var)

              MAXLEN(max-length)


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK ('LMGET',data-id

                     ,'MOVE  '|'LOCATE'|'INVAR '

                     ,dataloc-var

                     ,datalen-var

                     ,max-length);
```

buf-len

Specifies a fullword fixed binary integer containing the length of "buffer."

buffer

Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id

Specifies the data-id associated with the data set to be read. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

MOVE|LOCATE|INVAR

Specifies whether the data is to be moved, located, or stored. Any mode can be specified by a calling program function, with information being passed through the data location variable (see below). Only the INVAR mode can be used by a command dialog, with data being returned to the command in the data location variable.

dataloc-var

Specifies the name of the data location variable. In MOVE mode, the variable contains a binary virtual storage address, at which the data read by LMGET is to be stored. In LOCATE mode, the address of the data read by LMGET is placed in the data location variable. In INVAR mode, the data read by LMGET is itself placed in the data location variable. This parameter has a minimum length of one character and a maximum length of eight characters.

datalen-var

Specifies the name of the variable into which LMGET stores the actual length of the record read. The maximum length of this parameter is eight characters.

max-length

Specifies, in bytes, the maximum record length to be read. This parameter must be a non-zero positive numeric value. In MOVE mode, the value is the maximum number of bytes of data to be moved. In INVAR mode, the value is the maximum number of bytes of data to be stored in the data location variable. The value is not changed by LMGET in either mode. The parameter is not used in LOCATE mode.

If the max-length specification causes a DBCS character string to be divided in the middle, the result is unpredictable.

The following return codes are possible:

0 - Normal completion

8 - End-of-file condition (no message formatted)

10 - No ISPF library or data set is associated with the given data-id (that is, LMINIT has not been completed)

12 - Data set is not open or is not open for input
   - LMMFIND not done for a partitioned data set
   - Invalid parameter value

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue

Example: Invoke the LMGET service to read a record from the data set associated with the data-id in variable DDVAR, in INVAR mode, with LOCVAR as the data location variable, LENVAR as the actual record length variable, and 80 bytes as the maximum record length.

The three ways of invoking the LMGET service for this example are:

- Using ISPEXEC

```
      ISPEXEC LMGET DATAID(&DDVAR)  MODE(INVAR) DATALOC(LOCVAR)  +
                    DATALEN(LENVAR) MAXLEN(80)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
        LMGET DATAID(&DDVAR)  MODE(INVAR) DATALOC(LOCVAR)
              DATALEN(LENVAR) MAXLEN(80)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
        CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
        CALL ISPLINK('LMGET',DDVAR,'INVAR ','LOCVAR ','LENVAR ',80);
```

LMPUT

The LMPUT service writes one logical record to the data set associated with a given data-id. The first LMPUT writes the first logical record to the data set, and subsequent invocations write succeeding records. The LMINIT service and LMOPEN service with the OUTPUT option must be completed before LMPUT can be used.

If the data set is an ISPF library or MVS partitioned data set, the LMMADD or LMMREP service must be invoked after the last LMPUT to update the directory and to write the last physical record.

If the data set is sequential, the LMCLOSE service must be invoked after the last LMPUT to write the last physical record.

The LMPUT service writes records to a data file as is. That is, the LMPUT service does not pack data before writing it if the data is in unpacked format.

```
ISPEXEC LMPUT DATAID(data-id)

              MODE(INVAR|MOVE)

              DATALOC(dataloc-var)

              DATALEN(data-length)

              [NOBSCAN]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK ('LMPUT',data-id

                     ,'INVAR'|'MOVE '

                     ,dataloc-var

                     ,data-length

                     ,' '

                     ,['NOBSCAN']);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer

Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id

Specifies the data-id name associated with the data set into which the record is to be written. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

INVAR|MOVE

Specifies whether the data-location parameter variable contains the data (INVAR mode) or the address of the data (MOVE mode) to be written. Only the INVAR mode can be used by a command dialog.

dataloc-var

Specifies the name of a variable, which on entry to the LMPUT service, contains either the data to be written (INVAR mode), or the fullword binary virtual storage address of the data to be written (MOVE mode).

The value of the variable passed from a program function can be either the data record itself, or the address of the data record, but must coincide with the INVAR|MOVE specification above. If the variable was passed from a command function (CLIST), it should always contain the data record. The maximum length of this parameter is eight characters.

data-length

Specifies, in bytes, the length of the logical record to be written. The parameter must be a positive non-zero numeric value.

If the data-length specification causes a DBCS character string to be divided in the middle, the result is unpredictable.

NOBSCAN

Specifies the No Backscan option; no truncation of trailing blanks for records of variable length.

The following return codes are possible:

0 - Normal completion

10 - No data set associated with the given data-id
(that is, LMINIT has not been completed)

12 - Data set is not open or is not open for output
- Invalid parameter value

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue

Example: Invoke the LMPUT service to write a data record, with a length of 80 bytes, contained in variable DATAVAR into the data set associated with the data-id in variable DDVAR.

The three ways of invoking the LMPUT service for this example are:

- Using ISPEXEC

```
        ISPEXEC  LMPUT DATAID(&DDVAR)   MODE(INVAR)     +
                       DATALOC(DATAVAR) DATALEN(80)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
           LMPUT DATAID(&DDVAR)   MODE(INVAR)
                 DATALOC(DATAVAR) DATALEN(80)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
          CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
          CALL ISPLINK('LMPUT',DDVAR,'INVAR','DATAVAR ',80);
```

Note: When using null variables with the LMPUT service, the null variables must be defined to have a length greater than zero. Programs containing definitions of null variables must specify the VDEFINE service with the NOBSCAN option. Null variables defined in CLISTs should be initialized with the &STR built-in function.

CHAPTER 6. MEMBER MANIPULATION SERVICES

([illegible] integrated DATASETS)

The following member manipulation services are described in this chapter:

LMMLIST Creates a list of the names of all members in an ISPF library or concatenated set of ISPF libraries or MVS partitioned data set associated with a given data-id.

LMMFIND Finds a specified member of an ISPF library or concatenated set of ISPF libraries or MVS partitioned data set associated with a given data-id.

LMMADD Adds a member to the specified data set or ISPF library and updates the data set directory with information about the member to be added. This service may be for an ISPF library or an MVS partitioned data set.

LMMDEL Deletes all directory information associated with a given member from the specified data set directory. This service may be for an ISPF library or an MVS partitioned data set.

LMMREN Updates the directory to rename a member of a data set. This service may be for an ISPF library or an MVS partitioned data set.

LMMREP Updates the directory to replace a member of a given data set. This service may be for an ISPF library or an MVS partitioned data set.

LMMLIST

The LMMLIST service creates a list of the first occurrence of all the members in an ISPF library or concatenated set of libraries, or an MVS partitioned data set associated with the given data-id. This list is sorted by member name. By repeated use of LMMLIST, the dialog can have access to each member name in the member list. The LMINIT and LMOPEN services must be completed before LMMLIST can be used.

```
ISPEXEC LMMLIST DATAID(data-id)

                [OPTION(LIST|FREE)]

                [MEMBER(member-var)]

                [STATS(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK ('LMMLIST',data-id

                       ,['LIST'|'FREE']

                       ,[member-var]

                       ,['YES'|'NO ']);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id
Specifies the data-id associated with the data set for which the member list is to be created. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

LIST|FREE
Specifies whether the member names in the list are to be returned to the dialog, or the storage used by the member list to be freed when the dialog has finished processing the list.

- LIST

  The first time that LMMLIST is invoked with the LIST option (the default), the member list is created, and the name of the first member in the list is returned in member-var. If statistics information for the member is requested, the statistics are returned (see the STATS parameter).

  Subsequent invocations of LMMLIST with the LIST option return succeeding member names and their statistics (if requested) until the end of the list is reached, as indicated by the return code. At this point, the dialog should invoke LMMLIST with the FREE option.

- FREE

  The FREE option specifies that the virtual storage acquired to create the member list is to be freed. Each creation of a member list should be matched by an invocation of LMMLIST with the FREE option.

member-var

Specifies the name of the variable into which the name of the member used for positioning in the member list is specified, or the name of the next member in the list is to be stored. The maximum length of this parameter is eight characters.

When you invoke LMMLIST for the first time, the MEMBER variable is used for positioning within the member list. To position at the beginning, set the MEMBER variable to blanks. If the requested member is not found, the next member in the member list is returned.

YES|NO

Specifies whether ISPF statistics are to be returned with the member name. This parameter can be used only with the LIST option. NO is the default. See the description of the STATS parameter in the LMMFIND service for a description of this parameter.

The following return codes are possible:

```
 0 - LIST option:  Normal completion.  The member list is
                   available and the next member in the list
                   is returned in the member-var parameter.
   - FREE option:  Normal completion.  The member list is freed
                   successfully.

 4 - Empty member list

 8 - LIST option:  End of member list
   - FREE option:  Member list does not exist

10 - No data set is associated with the given data-id
     (that is, LMINIT has not been completed)
```

12 - Data set is not open or is not partitioned
   - Invalid parameter value

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue

Example:

Invoke the LMMLIST service with the LIST option to create a member list of the data set associated with the data-id in variable DDVAR, and to return the first member name in the list in variable MEMVAR.

The three ways of invoking the LMMLIST service for this example are:

- Using ISPEXEC
  ```
  ISPEXEC LMMLIST DATAID(&DDVAR) OPTION(LIST)        +
                  MEMBER(MEMVAR)
  ```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:
  ```
  LMMLIST DATAID(&DDVAR) OPTION(LIST)
          MEMBER(MEMVAR)
  ```
  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:
  ```
  CALL ISPEXEC (BUFLEN, BUFFER);
  ```

- Calling ISPLINK
  ```
  MEMVAR = ' ';
  CALL ISPLINK ('LMMLIST',DDVAR,'LIST','MEMVAR ');
  ```

LMMFIND

The LMMFIND service finds a specified member of an ISPF library or a partitioned data set associated with a given data-id. LMMFIND can also be used to return member statistics to the user. If the data-id represents a concatenated set of ISPF libraries, LMMFIND finds the first occurrence of the member in the set of libraries. The LMINIT and LMOPEN services must be completed before LMMFIND can be used.

```
ISPEXEC LMMFIND DATAID(data-id)

                MEMBER(member-name)

                [LOCK]

                [LRECL(lrecl-var)]

                [RECFM(recfm-var)]

                [GROUP(group-var)]

                [STATS(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK ('LMMFIND',data-id

                       ,member-name

                       ,['LOCK']

                       ,[lrecl-var]

                       ,[recfm-var]

                       ,[group-var]

                       ,['YES'|'NO ']);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer

Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id

Specifies the data-id associated with the data set to be searched. The data-id is generated by the LMINIT service. The maximum length of this parameter is eight characters.

member-name

Specifies the name of the member to be found. The maximum length of this parameter is eight characters.

LOCK

Indicates that the LMMFIND service is being used to find a member in a controlled ISPF library with the intention of updating it or with the intention of preventing others from updating it. This parameter can be used only if the data set is an ISPF library or concatenated set of ISPF libraries. The library management facility control file must have been allocated under the ddname ISRCFIL. For a further discussion of LOCK, see "Library Control Service" on page 78 and ISPF/PDF for MVS Library Management.

lrecl-var

Specifies the name of the variable into which the data record length (or,if the record format is of variable length, the maximum data record length) is to be stored. The maximum length of this parameter is eight characters.

recfm-var

Specifies the name of the variable into which the record format code (for example, 'FB' for fixed length block data) is to be stored. The maximum length of this parameter is eight characters.

group-var

Specifies the name of the variable into which the name of that group in which the member was found is to be stored. This variable contains the group name after the service is executed only if the data set is an ISPF library or a set of concatenated ISPF libraries and LMINIT is used with ISPF name parameters; otherwise, the variable is set to null. The maximum length of this parameter is eight characters.

YES|NO

Specifies whether statistics for the member are to be returned to the dialog invoking the service. If NO (the default) is specified, no statistics are returned. If YES is specified and the data-id represents a data set that has unformatted records (RECFM=U), only the concatenation set position is returned (ZLLIB), and all other statistical variables are not changed. For other record formats, the statistics are returned in the following dialog variables:

ZLLIB — Position in concatenated data set (a number from 1 to 4)

ZLVERS — Version number (a number from 1 to 99)

ZLMOD — Modification level (a number from 0 to 99)

ZLCDATE — Creation date. The creation date is a character value in the national format. For example: yy/mm/dd, mm/dd/yy, dd/mm/yy.

ZLMDATE — Last modification date. This date is in the national format and is the date that the statistics were saved. See "Creation date" above for an example of the date formats.

ZLMTIME — Last modification time (a character value in the format hh:mm)

ZLCNORC — Current number of records (a number from 0 to 65535)

ZLINORC — Initial number of records (a number from 0 to 65535)

ZLMNORC — Number of modified records (a number from 0 to 65535)

ZLUSER — Userid of last user to modify the given member (an alphameric field with a maximum length of eight characters)

Notes:

1. For an MVS partitioned data set, if the statistics are not stored in the data set directory in the same format used by ISPF/PDF Edit, only ZLLIB contains the concatenation set position. All other statistics variables are set to nulls.

2. These variables are stored into the function pool and are immediately available to command procedures. You cannot use the VGET service to retrieve these variables because the VGET service accesses the shared and profile variable pools.

The following return codes are possible:

0 - Normal completion

If LOCK is specified and the member is in a controlled ISPF library, a return code of zero indicates that the user is a valid user as specified by the member access ID, and that the member is available and is now currently owned by the user. A currently owned member is not available for updating by any other user. For more information on the member access ID, see ISPF/PDF for MVS Library Management.

```
 4 - Member not available:  in a controlled ISPF library and
     currently owned by another user
   - Member not available:  in a controlled ISPF library and
     user is not a valid user as specified by member access ID

 8 - Member not found

10 - No data set is associated with the given data-id
     (that is, LMINIT has not been completed)

12 - Data set is not open or is not open for input
     or is not partitioned
   - Invalid parameter value

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue
```

Example:

Invoke the LMMFIND service to locate member MYPROG in the data set associated with the data-id stored in DDVAR. Store the record length in variable LENVAR, the record format code in FORMVAR, and the name of the group in which it was found in GRPVAR.

The three ways of invoking the LMMFIND service for this example are:

- Using ISPEXEC

```
        ISPEXEC LMMFIND DATAID(&DDVAR) MEMBER(MYPROG)         +
                        LRECL(LENVAR)  RECFM(FORMVAR)         +
                        GROUP(GRPVAR)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
        LMMFIND DATAID(&DDVAR) MEMBER(MYPROG)
                LRECL(LENVAR)  RECFM(FORMVAR)
                GROUP(GRPVAR)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
        CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
        CALL ISPLINK ('LMMFIND',DDVAR,'MYPROG  ',' ',
                      'LENVAR ','FORMVAR ','GRPVAR ');
```

LMMADD

The LMMADD service adds a member to the specified data set or ISPF library. LMMADD then updates the data set directory with information about the member to be added. This service can be used for an ISPF library or an MVS partitioned data set. If the member already exists in the directory, the member name entry is not added to it. The LMINIT, LMOPEN with the OUTPUT option, and LMPUT services must be completed before LMMADD is used.

```
ISPEXEC LMMADD DATAID(data-id)

               MEMBER(member-name)

               [STATS(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK ('LMMADD',data-id

                      ,member-name

                      ,['YES'|'NO ']);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id
Specifies the data-id associated with the data set to which a member is being added. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

member-name
Specifies the member name being added to the directory of the data set. The maximum length of this parameter is eight characters.

YES|NO
Specifies whether the user data area in the directory should be updated so that the statistics of the member are stored in the

format used by ISPF/PDF Edit. If "YES" is specified, and the data set is partitioned and the record format is not unformatted (RECFM=U), the directory is updated with the member statistics. At least a valid creation date and the date of last modification must be provided in the member statistics. If "NO" (the default) is specified, statistics are not updated.

The following dialog variables are used to pass statistical information from the dialog invoking the LMMADD service:

ZLVERS Version number (a numeric value from 1 to 99)

ZLMOD Modification level (a number from 0 to 99)

ZLCDATE Creation date (a character value in the national format)

ZLMDATE Last modification date (a character value in the national format)

ZLMTIME Last modification time (a character value in the format hh:mm)

ZLCNORC Current number of records (a number from 0 to 65535)

ZLINORC Initial number of records (a number from 0 to 65535)

ZLMNORC Number of modified records (a number from 0 to 65535)

ZLUSER User id of the last user to modify the given member (must be a valid name, with a maximum length of eight characters)

The following return codes are possible:

0 - Normal completion

4 - Directory already contains the specified name

10 - No ISPF library or MVS data set is associated with the given data-id (that is, LMINIT has not been completed)

12 - Data set is not open or is not open for output
- Invalid parameter value
- Invalid data set organization
- Invalid values for some member statistics

14 - No record has been written for the member to be added

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue

Example:

Invoke the LMMADD service to add member MYPROG to the data set associated with the data-id in variable DDVAR.

The three ways of invoking the LMMADD service for this example are:

- Using ISPEXEC
  ```
  ISPEXEC LMMADD DATAID(&DDVAR) MEMBER(MYPROG)
  ```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:
  ```
  LMMADD DATAID(&DDVAR) MEMBER(MYPROG)
  ```
  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:
  ```
  CALL ISPEXEC (BUFLEN, BUFFER);
  ```

- Calling ISPLINK
  ```
  CALL ISPLINK('LMMADD',DDVAR,'MYPROG ');
  ```

LMMDEL

The LMMDEL service deletes all directory information associated with a given member from the specified data set directory. This service can be used for an ISPF library or an MVS partitioned data set. The LMINIT and LMOPEN (with the OUTPUT option) services must be completed before LMMDEL is used.

```
ISPEXEC LMMDEL DATAID(data-id)

               MEMBER(member-name)


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMMDEL',data-id

                     ,member-name);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id
Specifies the data-id associated with the data set from which a member is to be deleted. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

member-name
Specifies the name of the member to be deleted. The maximum length of this parameter is eight characters.

The following return codes are possible:

0 - Normal completion

8 - Member not found

10 - No data set is associated with the given data-id (that is, LMINIT has not been completed)

12 - Data set is not open or is not open for output
   - Invalid parameter value
   - Invalid data set organization

20 - Severe error, unable to continue

Example:

Invoke the LMMDEL service to delete member MYPROG from the data set associated with the data-id in variable DDVAR.

The three ways of invoking the LMMDEL service for this example are:

- Using ISPEXEC

```
        ISPEXEC LMMDEL DATAID(&DDVAR) MEMBER(MYPROG)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
        LMMDEL DATAID(&DDVAR) MEMBER(MYPROG)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
        CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
        CALL ISPLINK('LMMDEL',DDVAR,'MYPROG ');
```

LMMREN

The LMMREN service updates the directory to rename a member of a data set. This service can be used for an ISPF library or an MVS partitioned data set. The LMINIT and LMOPEN (with OUTPUT option) services must be completed before LMMREN can be used.

```
ISPEXEC LMMREN DATAID(data-id)

               MEMBER(old-member-name)

               NEWNAME(new-member-name)


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMMREN',data-id

                     ,old-member-name

                     ,new-member-name);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id
Specifies the data-id associated with the data set in which a member is being renamed. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

old-member-name
Specifies the present name of the member. The maximum length of this parameter is eight characters.

new-member-name
Specifies the new member name. The maximum length of this parameter is eight characters.

The following return codes are possible:

0 - Normal completion

4 - Directory already contains the specified new name

8 - Member not found

10 - No data set is associated with the given data-id (that is, LMINIT has not been completed)

12 - Data set is not open or is not open for output
- Invalid parameter value
- Invalid data set organization

20 - Severe error, unable to continue

Example:

Invoke the LMMREN service to rename member MYPROG (in the data set associated with the data-id in variable DDVAR) to MYPROGA.

The three ways of invoking the LMMREN service for this example are:

- Using ISPEXEC

```
        ISPEXEC LMMREN DATAID(&DDVAR)      +
                       MEMBER(MYPROG) NEWNAME(MYPROGA)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
        LMMREN DATAID(&DDVAR)
               MEMBER(MYPROG) NEWNAME(MYPROGA)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
        CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
        CALL ISPLINK('LMMREN',DDVAR,'MYPROG ','MYPROGA ');
```

LMMREP

The LMMREP service updates the directory to replace a member of a given data set. The specified member is added if it does not currently exist. This service can be used for an ISPF library or an MVS partitioned data set. The LMINIT, LMOPEN (with OUTPUT option), and LMPUT services must be completed before LMMREP can be used.

```
ISPEXEC LMMREP DATAID(data-id)

               MEMBER(member-name)

               [STATS(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMMREP',data-id

                     ,member-name

                     ,['YES'|'NO ']);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

data-id
Specifies the data-id associated with the data set in which a member is to be replaced. The data-id has been generated by the LMINIT service. The maximum length of this parameter is eight characters.

member-name
Specifies the name of the member to be replaced. The maximum length of this parameter is eight characters.

YES|NO
Specifies whether the user data area in the directory should be updated with the statistics of the member stored in the same format used by ISPF/PDF Edit. If "YES" is specified, and the data set is partitioned and the record format is not unformatted (RECFM=U), the

directory is updated with the member statistics. At least a valid creation date and the date of modification must be provided in the member statistics. If "NO" (the default) is specified, the statistics are not updated.

The following dialog variables are used to pass statistical information from the dialog invoking the LMMREP service:

ZLVERS Version number (a number from 1 to 99)

ZLMOD Modification level (a number from 0 to 99)

ZLCDATE Creation date (a character value in the national format)

ZLMDATE Last modification date (a character value in the national format)

ZLMTIME Last modification time (a character value in the format hh:mm)

ZLCNORC Current number of records (a number from 0 to 65535)

ZLINORC Initial number of records (a number from 0 to 65535)

ZLMNORC Number of modified records (a number from 0 to 65535)

ZLUSER User id of the last user to modify the given member (must be a valid name, with a maximum length of eight characters)

The following return codes are possible:

0 - Normal completion

8 - Member is added; it did not previously exist

10 - No data set is associated with the given data-id (that is, LMINIT has not been completed)

12 - Data set is not open or is not open for output
- Invalid parameter value
- Invalid data set organization
- Invalid values for some member statistics

14 - No record has been written for the member to be replaced

16 - Truncation or translation error in accessing dialog variables

20 - Severe error, unable to continue

Example:

Invoke the LMMREP service to update the directory of the data set associated with the data-id in variable DDVAR to replace member MYPROG.

The three ways of invoking the LMMREP service for this example are:

- Using ISPEXEC

  ```
  ISPEXEC LMMREP DATAID(&DDVAR) MEMBER(MYPROG)
  ```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

  ```
  LMMREP DATAID(&DDVAR) MEMBER(MYPROG)
  ```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

  ```
  CALL ISPEXEC (BUFLEN, BUFFER);
  ```

- Calling ISPLINK

  ```
  CALL ISPLINK('LMMREP',DDVAR,'MYPROG ');
  ```

CHAPTER 7. DATA SET SERVICES

The data set services described in this chapter are:

LMERASE Deletes an entire ISPF library or an MVS data set with a 3-level qualified data set name.

LMRENAME Renames an ISPF library or an MVS data set with a 3-level qualified data set name.

LMERASE

The LMERASE service deletes an entire ISPF library or an MVS data set with a 3-level qualified data set name. All members of the data set and all statistics are erased. The data set name used must be the cataloged name, not an alias data set name.

```
ISPEXEC LMERASE PROJECT(project)

                GROUP(group)

                TYPE(type)


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMERASE',project

                      ,group

                      ,type);
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

project
Indicates the highest level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. The maximum length of this parameter is eight characters.

group
Indicates the second-level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. The maximum length of this parameter is eight characters.

type
Indicates the third-level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. The maximum length of this parameter is eight characters.

The following return codes are possible:

0 - Normal completion

8 - Specified data set is not cataloged
  - Erase data set failed
  - Data set has not expired yet

20 - Severe error, unable to continue

Example:

Invoke LMERASE to delete a data set, with a 3-level qualified data set name, that has DEPT877 as its highest level qualifier, PRIVATE as its second level qualifier, and EXECS as its third-level qualifier.

The three ways of invoking the LMERASE function for this example are:

- Using ISPEXEC

```
        ISPEXEC LMERASE PROJECT(DEPT877)          +
                        GROUP(PRIVATE)            +
                        TYPE(EXECS)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
        LMERASE PROJECT(DEPT877)
                GROUP(PRIVATE)
                TYPE(EXECS)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
        CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
        CALL ISPLINK('LMERASE','DEPT877 ',
                               'PRIVATE ',
                               'EXECS ');
```

LMRENAME

The LMRENAME service renames an ISPF library or an MVS data set with a 3-level qualified data set name. The data set name used must be the cataloged name, not an alias data set name.

```
ISPEXEC LMRENAME PROJECT(project)

                 GROUP(group)

                 TYPE(type)

                 {[NEWPROJ(new-project)]
                  [NEWGROUP(new-group)]
                  [NEWTYPE(new-type)]    }


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMRENAME',project

                       ,group

                       ,type

                       {,[new-project]
                         ,[new-group]
                         ,[new-type]    });
```

buf-len
Specifies a fullword fixed binary integer containing the length of "buffer."

buffer
Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

project
Indicates the highest level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. The maximum length of this parameter is eight characters.

group
Indicates the second-level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. The maximum length of this parameter is eight characters.

type
Indicates the third-level qualifier in the specification of an ISPF library or an MVS data set with a 3-level qualified data set name. The maximum length of this parameter is eight characters.

new-project
Indicates the new highest level qualifier. If this parameter is not specified, the project parameter value is used. The maximum length of this parameter is eight characters.

new-group
Indicates the new second-level qualifier. If this parameter is not specified, the group parameter value is used. The maximum length of this parameter is eight characters.

new-type
Indicates the new third-level qualifier. If this parameter is not specified, the type parameter value is used. The maximum length of this parameter is eight characters.

Note: Either new-project, new-group, or new-type must be specified.

The following return codes are possible:

0 - Normal completion

4 - New name already exists

8 - Specified data set does not exist
- Rename or catalog failed
- Alias data set names not permitted

12 - Invalid parameter value

20 - Severe error, unable to continue

Example:

Invoke the LMRENAME service to rename a data set with the name DEPT877.PRIVATE.ASSEMBLE to DEPT877.MINE.ASSEMBLE.

The three invocations of the LMRENAME service for this example are given on the next page.

- Using ISPEXEC

```
        ISPEXEC LMRENAME PROJECT(DEPT877)     +
                         GROUP(PRIVATE)       +
                         TYPE(ASSEMBLE)       +
                         NEWGROUP(MINE)
```

- Calling ISPEXEC
  Set the program variable BUFFER to contain:

```
        LMRENAME PROJECT(DEPT877)
                 GROUP(PRIVATE)
                 TYPE(ASSEMBLE)
                 NEWGROUP(MINE)
```

  Set program variable BUFLEN to the length of the variable BUFFER.
  Issue the following:

```
        CALL ISPEXEC (BUFLEN, BUFFER);
```

- Calling ISPLINK

```
        CALL ISPLINK('LMRENAME','DEPT877 ',
                                'PRIVATE ',
                                'ASSEMBLE',' ',
                                'MINE ');
```

CHAPTER 8. LIBRARY CONTROL SERVICE

When the LMMFIND service is used to find a member in a controlled ISPF library to update that member, the parameter LOCK is specified. When the member is accessed, it is locked, that is, made unavailable for updating by any other person. The member becomes available to other people when it is promoted to the controlled library or LMPROM is used with the FREE parameter specified.

The library control service described in this chapter is:

LMPROM Promotes a member or sequential data set to a controlled ISPF library or frees a locked member in a controlled ISPF library.

LMPROM

The LMPROM service is used to move a member or data set in an "uncontrolled" library to a "controlled" ISPF entry-level library. It also promotes members between the "controlled" ISPF libraries according to the promotion hierarchy information in the library management facility control file. (See ISPF/PDF for MVS Library Management for more information on the library management facility.)

The prerequisites for using the LMPROM service are:

1. The library management facility control file must have been allocated under the ddname ISRCFIL.

2. The promotion hierarchy must have been created and activated by the library administrator using the ISPF/PDF library management utilities.

3. The user must be authorized to promote the members in the promotion hierarchy.

```
ISPEXEC LMPROM {FRPROJ(from-project) FRGROUP(from-group)
                   FRTYPE(from-type) FRMEMBER(from-member)}

               {DATASET(dsname)                            }

               [TOPROJ(to-project)]    [TOGROUP(to-group)]

               [TOTYPE(to-type)]    [TOMEMBER(to-member)]

               [VOLUME(serial)]    [PASSWORD(password)]

               [ACTION(MOVE|FREE)]    [REASON(reason-code)]

               [PACK(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMPROM'  ,{from-project,from-group
                         ,from-type,from-member    }

                        ,{dsname                   }

                        ,[serial] ,[password]

                        ,['MOVE'|'FREE'] ,[reason-code]

                        ,['YES'|'NO ']

                        ,[to-project] ,[to-group]

                        ,[to-type] ,[to-member]);
```

Either the 3-level qualified name or the dsname of the "from" data set must be specified. LMPROM looks first to see whether the dsname is specified. If so, LMPROM uses the dsname as the data set name. If dsname is not specified, LMPROM uses the 3-level qualified name as the data set name. If neither the 3-level qualified name nor the dsname is specified, an error message is issued.

If the promotion is from a "controlled" ISPF library to a "controlled" ISPF library, the "to" ISPF library qualifier must not be specified. If the qualifier is specified, a message appears.

buf-len

Specifies a fullword fixed binary integer containing the length of "buffer."

buffer

Specifies a buffer containing the name of the service and its parameters in the same form as they would appear in an ISPEXEC invocation for a command procedure.

from-project

Indicates the highest-level qualifier in the name of the ISPF library or MVS data set with a 3-level qualified data set name from which the member is being promoted. This parameter is required if the dsname is not specified. The maximum length of this parameter is eight characters.

from-group

Indicates the second-level qualifier in the name of the ISPF library or MVS data set with a 3-level qualified data set name from which the member is being promoted. This parameter is required if the dsname is not specified. The maximum length of this parameter is eight characters.

from-type

Indicates the third-level qualifier in the name of the ISPF library or MVS data set with a 3-level qualified data set name from which the member is being promoted. This parameter is required if the dsname is not specified. The maximum length of this parameter is eight characters.

from-member

Specifies the name of the member being promoted from the ISPF library or MVS data set with a 3-level qualified data set name. This parameter is required if the dsname is not specified. The maximum length of this parameter is eight characters.

dsname

Specifies the name of the partitioned or sequential data set being promoted. The specification of a partitioned data set must include a member name in parentheses. The maximum length of this parameter is 56 characters.

serial

Specifies the volume serial number of the DASD volume on which the data set resides. This parameter is associated with the dsname parameter above, but is required only if the data set is not cataloged. The maximum length of this parameter is six characters.

password

Specifies the MVS password of the data set. This parameter is required only if the data set is password-protected. (The password is not specified for RACF or PCF protected data sets.) The maximum length of this parameter is eight characters.

MOVE|FREE

Specifies the options of the LMPROM service.

The LMPROM service with MOVE (the default option) promotes a member from a data set to a "controlled" ISPF library. Thereafter, the member is available for update by other authorized owners. After the LMPROM service is completed, the "from" member or the "from" data set is deleted. If a zero length member is promoted to a "controlled" ISPF library that is at the top of a promotion hierarchy or if the ISPF library is the only library within the promotion hierarchy, that member and its controls are deleted.

The LMPROM service with FREE makes the named member available for updating by other authorized owners. The FREE option is only valid for promoting a member from an "uncontrolled" ISPF library to a "controlled" ISPF library. No "move" is actually done.

reason-code

Specifies the reason for the promotion. The maximum length of this parameter is 26 characters. The code cannot contain blanks, commas, semicolons, or parentheses. If several reason codes are to be entered, they can be delimited by other characters, such as periods or slashes. This is used only when promoting to an entry-level library or when using the FREE parameter.

YES|NO

Specifies whether the member being promoted is to be packed in the "controlled" library. If the keyword is not specified, the library controls defined for the "controlled" library are used. This is used only when promoting to an entry-level library.

PACK uses a unique ISPF data compression routine. Attempts to access or execute the data outside of PDF can cause unpredictable results. See the description of the Edit PACK command in ISPF/PDF for MVS Program Reference for more information.

to-project

Indicates the highest-level qualifier in the name of the "controlled" ISPF library (an entry-level library) to which the member is being promoted. If the from-project is specified but the to-project is not specified, the from-project value is used for both the "to" and "from" project parameters. If the promotion is from a "controlled" ISPF library to a "controlled" ISPF library, the to-project parameter must not be used. However, the to-project parameter is required if the dsname is specified. The maximum length of this parameter is eight characters.

to-group

Indicates the second-level qualifier in the name of the "controlled" ISPF library (an entry-level library) to which the member is being promoted. This parameter is required whenever the to-project parameter is required. The to-group parameter must not be used if the "from" data set is a "controlled" ISPF library. The maximum length of this parameter is eight characters.

to-type

Indicates the third-level qualifier in the name of the "controlled" ISPF library to which the member is being promoted. This parameter is required if the dsname is specified. If to-type is not specified, the from-type value is used for both the "to" and "from" parameters. The to-type parameter must not be used if the "from" data set is a "controlled" ISPF library. The maximum length of this parameter is eight characters.

to-member

Specifies the name of the member being promoted into the "controlled" ISPF library. If to-member is not specified, the from-member value is used for both the "to" and "from" parameters. This parameter is required when the "from" data set is sequential. The to-member parameter must not be used if the "from" data set is a "controlled" ISPF library. The maximum length of this parameter is eight characters.

The following return codes are possible:

0 - Normal completion

8 - Promote failed; the error condition is described in ISPF system dialog variables

12 - Invalid parameter value

20 - Severe error, unable to continue

Example:

Invoke the LMPROM service to promote member UTILITY from the "uncontrolled" library ISPF.MYLIB.PANELS to the "controlled" entry-level library ISPF.ISPFLIB.PANELS with reason code PTM3456.

- Using ISPEXEC

  ```
  ISPEXEC LMPROM FRPROJ(ISPF) FRGROUP(MYLIB) FRTYPE(PANELS) +
                 FRMEMBER(UTILITY) TOGROUP(ISPFLIB)        +
                 REASON(PTM3456)
  ```

- Calling ISPEXEC

  Set the program variable BUFFER to contain:

  ```
  LMPROM FRPROJ(ISPF) FRGROUP(MYLIB) FRTYPE(PANELS)
         FRMEMBER(UTILITY) TOGROUP(ISPFLIB)
         REASON(PTM3456)
  ```

  Set program variable BUFLEN to the length of the variable BUFFER. Issue the following:

  ```
  CALL ISPEXEC (BUFLEN, BUFFER);
  ```

- Calling ISPLINK

  ```
  CALL ISPLINK('LMPROM','ISPF    ','MYLIB   ','PANELS  '
         ,'UTILITY ',' ' ,' ' ,' ' ,' ' ,'PTM3456 '
         ,' ' ,' ' ,'ISPFLIB ');
  ```

APPENDIX A. LIST OF PDF SERVICES

The following is an alphabetized summary of the PDF services, with the page on which details for each service can be found.

BROWSE - Provides an interface to the ISPF/PDF Browse option. Page 26

```
ISPEXEC BROWSE DATASET(dsname) [VOLUME(serial)]

                               [PASSWORD(pswd-value)]

                               [PANEL(panel-name)]

                               [FORMAT(format-name)]

                               [YES|NO]
    or

ISPEXEC BROWSE DATAID(data-id) [MEMBER(member-name)]

                               [PANEL(panel-name)]

                               [FORMAT(format-name)]

                               [YES|NO]

CALL ISPEXEC (buf–len,buffer);

CALL ISPLINK ('BROWSE',{dsname} ,[serial]

                         ,[pswd–value]

                         ,[panel–name]

                        {data-id}

                         ,[member–name]

                         ,[format-name]

                         ,['YES'|'NO ']);
```

EDIT - Provides an interface to the ISPF/PDF editor   Page 30

```
ISPEXEC EDIT DATASET(dsname) [VOLUME(serial)]

                             [PASSWORD(pswd-value)]

                             [PANEL(panel-name)]

                             [MACRO(macro-name)]

                             [PROFILE(profile-name)]

                             [FORMAT(format-name)]

                             [YES|NO]

or

ISPEXEC EDIT DATAID(data-id) [MEMBER(member-name)]

                             [PANEL(panel-name)]

                             [MACRO(macro-name)]

                             [PROFILE(profile-name)]

                             [FORMAT(format-name)]

                             [YES|NO]

CALL ISPEXEC (buf-len,buffer);

CALL ISPLINK ('EDIT',{dsname} ,[serial]

                      ,[pswd-value] ,[panel-name]

                      ,[macro-name] ,[profile-name]

                     {data-id}

                      ,[member-name]

                      ,[format-name]

                      ,['YES'|'NO ']);
```

EDREC - Initializes an edit recovery table Page 35

```
ISPEXEC EDREC {INIT [CMD(command-name)]}

              {QUERY}

              {PROCESS [PASSWORD(pswd-value)] [DATAID(data-id)]}

              {CANCEL}

              {DEFER}

CALL ISPEXEC (buf-len,buffer);

CALL ISPLINK ('EDREC' ,{'INIT    ' ,[command-name]}

                      ,{'QUERY   '}

                      ,{'PROCESS' ,[pswd-value] ,[data-id]}

                      ,{'CANCEL '}

                      ,{'DEFER   '});
```

LMCLOSE - Closes a data set Page 43

```
ISPEXEC LMCLOSE DATAID(data-id)

CALL ISPEXEC (buf-len, buffer);

CALL ISPLINK('LMCLOSE',data-id);
```

LMERASE - Deletes an ISPF library Page 72
or MVS data set

```
ISPEXEC LMERASE PROJECT(project)

                GROUP(group)

                TYPE(type)

CALL ISPEXEC (buf-len, buffer);

CALL ISPLINK('LMERASE',project

                      ,group

                      ,type);
```

LMFREE - Removes the data-id and releases the data set associated with a given data-id Page 22

```
ISPEXEC LMFREE DATAID(data-id)

CALL ISPEXEC (buf-len, buffer);

CALL ISPLINK('LMFREE',data-id);
```

LMGET - Reads one record of a sequential data set or of a member of a partitioned data set identified by the given data id. Page 46

```
ISPEXEC LMGET DATAID(data-id)

                  MODE(MOVE|LOCATE|INVAR)

                  DATALOC(dataloc-var)

                  DATALEN(datalen-var)

                  MAXLEN(max-length)

CALL ISPEXEC (buf-len, buffer);

CALL ISPLINK('LMGET',data-id

                          ,'MOVE  '|'LOCATE'|'INVAR '

                          ,dataloc-var

                          ,datalen-var

                          ,max-length);
```

LMINIT - Associates a data-id with a specified data set

Page 12

```
ISPEXEC LMINIT DATAID(data-id-var)

                 {PROJECT(project) GROUP1(group1) TYPE(type)   }
                  [GROUP2(group2)] [GROUP3(group3)]
                  [GROUP4(group4)]

                 {DATASET(dsname)                              }

                 {DDNAME(ddname)                               }

                 [VOLUME(serial)] [PASSWORD(password)]

                 [ENQ(SHR|EXCLU|SHRW|MOD)]

                 [ORG(org-var)]
```

```
CALL ISPEXEC (buf-len,buffer);
```

```
CALL ISPLINK ('LMINIT', data-id-var

                           ,{project,group1,type     }
                             ,[group2] ,[group3]
                             ,[group4]

                           {dsname                   }

                           {ddname                   }

                           ,[serial] ,[password]

                           ,['SHR  '|'EXCLU'|'SHRW '|'MOD  ']

                           ,[org-var]);
```

LMMADD - Adds a member to an ISPF library or a partitioned data set   Page 61

```
ISPEXEC LMMADD DATAID(data-id)

               MEMBER(member-name)

               [STATS(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMMADD',data-id

                     ,member-name

                     ,['YES'|'NO ']);
```

LMMDEL - Deletes a member of an ISPF library or a partitioned data set   Page 64

```
ISPEXEC LMMDEL DATAID(data-id)

               MEMBER(member-name)


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMMDEL',data-id

                     ,member-name);
```

LMMFIND - Finds a member of an ISPF library or partitioned data set   Page 57

```
ISPEXEC LMMFIND DATAID(data-id)

                MEMBER(member-name)

                [LOCK]

                [LRECL(lrecl-var)]

                [RECFM(recfm-var)]

                [GROUP(group-var)]

                [STATS(YES|NO)]


CALL ISPEXEC (buf-len, buffer);

CALL ISPLINK('LMMFIND',data-id

                      ,member-name

                      ,['LOCK']

                      ,[lrecl-var]

                      ,[recfm-var]

                      ,[group-var]

                      ,['YES'|'NO ']);
```

LMMLIST - Creates a member list of an ISPF library or partitioned data set Page 54

```
ISPEXEC LMMLIST DATAID(data-id)

                [OPTION(LIST|FREE)]

                [MEMBER(member-var)]

                [STATS(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMMLIST',data-id

                      ,['LIST'|'FREE']

                      ,[member-var]

                      ,['YES'|'NO ']);
```

LMMREN - Renames a member of an ISPF library or partitioned data set Page 66

```
ISPEXEC LMMREN DATAID(data-id)

               MEMBER(old-member-name)

               NEWNAME(new-member-name)


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMMREN',data-id

                     ,old-member-name

                     ,new-member-name);
```

LMMREP - Replaces a member of an
ISPF library or
partitioned data set   Page 68

```
ISPEXEC LMMREP DATAID(data-id)

               MEMBER(member-name)

               [STATS(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMMREP',data-id

                      ,member-name

                      ,['YES'|'NO ']);
```

LMOPEN - Opens a data set   Page 40

```
ISPEXEC LMOPEN DATAID(data-id)

               [OPTION(INPUT|OUTPUT)]

               [LRECL(lrecl-var)]

               [RECFM(recfm-var)]

               [ORG(org-var)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMOPEN ',data-id

                       ,['INPUT '|'OUTPUT']

                       ,[lrecl-var]

                       ,[recfm-var]

                       ,[org-var]);
```

LMPROM - Promotes a member into a controlled ISPF library   Page 77

```
ISPEXEC LMPROM {FRPROJ(from-project) FRGROUP(from-group)
                    FRTYPE(from-type) FRMEMBER(from-member)}

               {DATASET(dsname)                            }

               [TOPROJ(to-project)]   [TOGROUP(to-group)]

               [TOTYPE(to-type)]   [TOMEMBER(to-member)]

               [VOLUME(serial)]   [PASSWORD(password)]

               [ACTION(MOVE|FREE)]   [REASON(reason-code)]

               [PACK(YES|NO)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMPROM' ,{from-project,from-group
                        ,from-type,from-member   }

                      ,{dsname)                  }

                      ,[serial]  ,[password]

                      ,['MOVE'|'FREE']  ,[reason-code]

                      ,['YES'|'NO ']

                      ,[to-project]  ,[to-group]

                      ,[to-type]  ,[to-member]);
```

LMPUT - Writes one record of a data set   Page 49

```
ISPEXEC LMPUT DATAID(data-id)

              MODE(INVAR|MOVE)

              DATALOC(dataloc-var)

              DATALEN(data-length)

              [NOBSCAN]

CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMPUT',data-id

                    ,'INVAR'|'MOVE '

                    ,dataloc-var

                    ,data-length

                    ,' '

                    ,['NOBSCAN']);
```

LMQUERY - Provides requested information regarding a data set associated with a given data-id

Page 19

```
ISPEXEC LMQUERY DATAID(data-id)

                [PROJECT(proj-var)]  [GROUP1(group1-var)]

                [GROUP2(group2-var)]  [GROUP3(group3-var)]

                [GROUP4(group4-var)]  [TYPE(type-var)]

                [DATASET(dsn-var)]  [DDNAME(ddn-var)]

                [VOLUME(serial-var)]  [ENQ(enq-var)]


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK ('LMQUERY',data-id

                       ,[proj-var] ,[group1-var]

                       ,[group2-var] ,[group3-var]

                       ,[group4-var] ,[type-var]

                       ,[dsn-var] ,[ddn-var]

                       ,[serial-var]

                       ,[' ',' ',' ',' ',enq-var]);
```

LMRENAME - Renames an ISPF library   Page 74

```
ISPEXEC LMRENAME PROJECT(project)

                 GROUP(group)

                 TYPE(type)

                 {[NEWPROJ(new-project)]
                  [NEWGROUP(new-group)]
                  [NEWTYPE(new-type)]   }


CALL ISPEXEC (buf-len, buffer);


CALL ISPLINK('LMRENAME',project

                       ,group

                       ,type

                       {,[new-project]
                        ,[new-group]
                        ,[new-type]   });
```

APPENDIX B. EXAMPLES

Example 1: Using Library Access Services

```
/******************************************************/
/*                                                    */
/*  THE OBJECTIVE OF THIS CLIST IS TO COPY A MEMBER   */
/*  FROM ONE ISPF LIBRARY TO ANOTHER.                 */
/*                                                    */
/******************************************************/

/******************************************************/
/*                                                    */
/*  INITIALIZATION                                    */
/*                                                    */
/******************************************************/

/******************************************************/
/*                                                    */
/*  INITIALIZE INPUT DATA SET NAMES                   */
/*                                                    */
/******************************************************/

SET &INPROJ   = ISPFPROJ    /* SET THE PROJECT NAME   */
SET &INGROUP  = OLDLIB      /* SET THE GROUP NAME     */
SET &INTYPE   = OLDTYPE     /* SET THE TYPE NAME      */
SET &INMEMB   = OLDMEMB     /* SET THE MEMBER NAME    */

/******************************************************/
/*                                                    */
/*  INITIALIZE OUTPUT DATA SET NAMES                  */
/*                                                    */
/******************************************************/

SET &OUTPROJ  = ISPFPROJ    /* SET THE PROJECT NAME   */
SET &OUTGROUP = NEWLIB      /* SET THE GROUP NAME     */
SET &OUTTYPE  = NEWTYPE     /* SET THE TYPE NAME      */
SET &OUTMEMB  = NEWMEMB     /* SET THE MEMBER NAME    */


SET &ON =1                  /* SET ON TO 1             */
SET &OFF =0                 /* SET OFF TO 0            */
SET &INITINP = &OFF         /* INPUT NOT ALLOC         */
SET &INITOUT = &OFF         /* OUTPUT NOT ALLOC        */
SET &OPENINP = &OFF         /* INPUT NOT OPEN          */
SET &OPENOUT = &OFF         /* OUTPUT NOT OPEN         */
SET &LMRC = 0               /* INITIALIZE RETURN CODE  */
```

```
/********************************************************/
/*                                                      */
/*  CALL LMINIT TO ALLOCATE INPUT DATA SET              */
/*                                                      */
/********************************************************/

ISPEXEC LMINIT DATASET('&INPROJ..&INGROUP..&INTYPE') +
               DATAID(INP) ENQ(SHR)
SET &RC = &LASTCC           /* GET THE RC FROM LMINIT  */
IF &RC ¬=  0 THEN -
  DO                        /* WRITE ERROR MESSAGE     */
    WRITE FAIL TO ALLOCATE INPUT DATA SET
    WRITE ERROR CONDITION IS &ZERRLM
  END
ELSE -
  SET &INITINP = &ON        /* ALLOCATE SUCCESSFULLY   */

/********************************************************/
/*                                                      */
/*  CALL  LMOPEN TO OPEN INPUT DATA SET                 */
/*                                                      */
/********************************************************/

IF &RC = 0 THEN  -
  DO
    ISPEXEC LMOPEN DATAID(&INP) OPTION(INPUT)
    SET &RC = &LASTCC       /* GET THE RC FROM LMOPEN  */
    IF &RC ¬=  0 THEN -
      DO                    /* WRITE ERROR MESSAGE     */
        WRITE FAIL TO OPEN INPUT DATA SET
        WRITE ERROR CONDITION IS &ZERRLM
      END
    ELSE -
      SET &OPENINP = &ON    /* OPEN SUCCESSFULLY       */
  END

/********************************************************/
/*                                                      */
/*  CALL  LMMFIND TO FIND THE INPUT MEMBER              */
/*                                                      */
/********************************************************/

IF &RC = 0 THEN  -
  DO
    ISPEXEC LMMFIND DATAID(&INP) MEMBER(&INMEMB)
    SET &RC = &LASTCC       /* GET THE RC FROM LMMFIND  */
    IF &RC ¬=  0 THEN  -
      DO                    /* WRITE ERROR MESSAGE      */
        WRITE FAIL TO FIND INPUT MEMBER
        WRITE ERROR CONDITION IS &ZERRLM
      END
  END
```

```
/********************************************************/
/*                                                      */
/*  CALL LMINIT TO ALLOCATE OUTPUT DATA SET             */
/*                                                      */
/********************************************************/

  IF &RC = 0 THEN  -
    DO
      ISPEXEC LMINIT DATASET('&OUTPROJ..&OUTGROUP..&OUTTYPE') +
                     DATAID(OUT) ENQ(EXCLU)
      SET &RC = &LASTCC     /* GET THE RC FROM LMINIT   */
      IF &RC ¬=  0 THEN -
        DO                  /* WRITE ERROR MESSAGE      */
          WRITE FAIL TO ALLOCATE OUTPUT DATA SET
          WRITE ERROR CONDITION IS &ZERRLM
        END
      ELSE -
        SET &INITOUT = &ON  /* ALLOCATE SUCCESSFULLY    */
    END

/********************************************************/
/*                                                      */
/*  CALL LMOPEN TO OPEN OUTPUT DATA SET                 */
/*                                                      */
/********************************************************/

  IF &RC = 0 THEN  -
    DO
        ISPEXEC LMOPEN DATAID(&OUT) OPTION(OUTPUT)
        SET &RC = &LASTCC   /* GET THE RC FROM LMOPEN   */
        IF &RC ¬= 0 THEN -
          DO                /* WRITE ERROR MESSAGE      */
            WRITE FAIL TO OPEN OUTPUT DATA SET
            WRITE ERROR CONDITION IS &ZERRLM
          END
        ELSE -
          SET &OPENOUT=&ON  /* OPEN SUCCESSFULLY        */
    END
```

```
/*******************************************************/
/*                                                     */
/*  COPY RECORDS FROM INPUT TO OUTPUT                  */
/*                                                     */
/*******************************************************/

 DO WHILE &LMRC=0 AND &RC=0 /* DATA SETS ARE OPEN AND  */
                            /* THERE IS NO I/O ERROR   */

   /****************************************************/
   /*                                                  */
   /*  GET A LOGICAL RECORD FROM INPUT DATA SET        */
   /*                                                  */
   /****************************************************/

   ISPEXEC LMGET DATAID(&INP) MODE(INVAR) DATALOC(RECLOC) +
                 DATALEN(RECLEN) MAXLEN(80)
   SET &LMRC = &LASTCC     /* GET THE RC FROM LMGET    */
   IF &LMRC = 0 THEN -
     DO

       /************************************************/
       /*                                              */
       /*  WRITE A LOGICAL RECORD TO OUTPUT DATA SET   */
       /*                                              */
       /************************************************/
       WRITE LOGICAL RECORD READ IS &RECLOC
       ISPEXEC LMPUT DATAID(&OUT) MODE(INVAR) +
                     DATALOC(RECLOC) DATALEN(&RECLEN) NOBSCAN
       SET &LMRC=&LASTCC   /* GET THE RC FROM LMPUT    */
       IF &LMRC ¬= 0 THEN -
         DO                /* WRITE ERROR MESSAGE      */
           WRITE FAIL TO PUT A LOGICAL RECORD TO OUTPUT DATA SET
           WRITE ERROR CONDITION IS &ZERRLM
         END
     END
 END
```

```
/*******************************************************/
/*                                                     */
/*  END OF FILE OR ERROR ENCOUNTERED ?                 */
/*                                                     */
/*******************************************************/

IF &RC = 0 AND &LMRC = 8 THEN -
  DO

  /*****************************************************/
  /*                                                   */
  /*  END OF FILE REACHED                              */
  /*  CALL LMMREP TO REPLACE OUTPUT MEMBER             */
  /*                                                   */
  /*****************************************************/

     ISPEXEC LMMREP DATAID(&OUT) MEMBER(&OUTMEMB)
     SET &RC = &LASTCC    /* GET THE RC FROM LMMREP    */
     IF &RC ¬= 0 AND &RC ¬= 8 THEN -
       DO                 /* WRITE ERROR MESSAGE       */
         WRITE FAIL TO REPLACE OUTPUT MEMBER
         WRITE ERROR CONDITION IS &ZERRLM
       END
   END
 ELSE -
   IF &RC = 0 AND &LMRC ¬= 0 THEN -
     DO                   /* WRITE ERROR MESSAGE       */
       WRITE FAIL TO GET A LOGICAL RECORD FROM INPUT MEMBER
       WRITE ERROR CONDITION IS &ZERRLM
     END
```

```
/*******************************************************/
/*                                                     */
/*  CALL LMCLOSE TO CLOSE INPUT DATA SET               */
/*                                                     */
/*******************************************************/

IF &OPENINP = &ON THEN  -
  DO                      /* INPUT DATA IS OPEN        */
    ISPEXEC LMCLOSE DATAID(&INP)    /* CLOSE IT        */
    SET &RC = &LASTCC     /* GET THE RC FROM LMCLOSE   */
    IF &RC = 0 THEN -
      DO                  /* CLOSE SUCCESSFULLY        */
        SET &OPENINP=&OFF /* INPUT DATA SET CLOSED     */
        IF &INITINP = &ON THEN -

        /***********************************************/
        /*                                             */
        /*  CALL LMFREE TO FREE INPUT DATA SET         */
        /*                                             */
        /***********************************************/

          DO
            ISPEXEC LMFREE DATAID(&INP) /* FREE INPUT  */
            SET &RC = &LASTCC  /* GET RETURN CODE      */
            IF &RC ¬= 0 THEN -
              DO          /* WRITE ERROR MESSAGE       */
                WRITE FAIL TO FREE INPUT DATA SET
                WRITE ERROR CONDITION IS &ZERRLM
              END
            ELSE SET &INITINP = &OFF /* INPUT NOT FREE */
          END
      END
     ELSE -
       DO                 /* WRITE ERROR MESSAGE       */
         WRITE FAIL TO CLOSE INPUT DATA SET
         WRITE ERROR CONDITION IS &ZERRLM
       END
   END
```

```
/*******************************************************/
/*                                                     */
/*  CALL LMCLOSE TO CLOSE OUTPUT DATA SET              */
/*                                                     */
/*******************************************************/

IF &OPENOUT = &ON THEN -
  DO                      /* OUTPUT OPEN ?             */
    ISPEXEC LMCLOSE DATAID(&OUT) /* CLOSE OUTPUT       */
    SET &RC = &LASTCC     /* GET THE RC FROM LMCLOSE   */
    IF &RC = 0  THEN -
      DO                  /* CLOSE SUCCESSFULLY        */
        SET &OPENOUT = &OFF  /* SET OUTPUT CLOSED      */
        IF &INITOUT = &ON THEN -
          DO              /* OUTPUT ALLOCATED ?        */

          /*********************************************/
          /*                                           */
          /*  CALL LMFREE TO FREE OUTPUT DATA SET      */
          /*                                           */
          /*********************************************/

            ISPEXEC LMFREE DATAID(&OUT) /* FREE OUTPUT */
            SET &RC = &LASTCC /* GET THE RETURN CODE   */
            IF &RC ¬= 0 THEN -
              DO          /* WRITE ERROR MESSAGE       */
                WRITE FAIL TO FREE OUTPUT DATA SET
                WRITE ERROR CONDITION IS &ZERRLM
              END
            ELSE SET &INITOUT = &OFF /* SET FLAG OFF   */
          END
      END
    ELSE -
      DO                  /* WRITE THE ERROR MESSAGE   */
        WRITE FAIL TO CLOSE OUTPUT DATA SET
        WRITE ERROR CONDITION IS &ZERRLM
      END
  END
 EXIT CODE(&RC)
```

Example 2: Using the ddname Interface to Browse and Edit

BROWSE

```
PROC 0
  /**********************************************************/
  /* EXAMPLE OF FILE TAILORING INTO A TEMP DATASET AND      */
  /* SUBSEQUENT BROWSE OF THAT TEMP DATASET.                */
  /**********************************************************/
  ISPEXEC FTOPEN TEMP              /* USE TEMP FOR TAILORING*/
  IF &LASTCC ¬= 0 THEN WRITE FTOPEN FAILED
  ISPEXEC FTINCL ISPPFTS1          /* INCLUDE EXAMPLE JCL   */
  IF &LASTCC ¬= 0 THEN WRITE FTINCL FAILED
  ISPEXEC FTCLOSE
  SET &ZTEMPN  =
  SET &DATA1 =
  ISPEXEC VGET (ZTEMPN) SHARED     /* GET DDNAME OF TEMP    */
  ISPEXEC LMINIT DATAID(DATA1) DDNAME(&ZTEMPN) ENQ(EXCLU)
  IF &LASTCC ¬= 0 THEN WRITE LMINIT FAILED
  ELSE +
    DO
      ISPEXEC BROWSE DATAID(&DATA1)
      ISPEXEC LMFREE DATAID(&DATA1)
    END
END
```

EDIT

```
PROC 0
  /**********************************************************/
  /* EXAMPLE OF FILE TAILORING INTO A VIO DATASET AND       */
  /* SUBSEQUENT EDIT OF THAT VIO DATASET.                   */
  /**********************************************************/
  ALLOC FI(ISPFILE) UNIT(VIO) CYL SPACE(1,1) LRECL(80)   +
        BLKSIZE(8000) RECFM(F,B) REUSE NEW
  ISPEXEC FTOPEN
  IF &LASTCC ¬= 0 THEN WRITE FTOPEN FAILED
  ISPEXEC FTINCL ISPPFTS1
  IF &LASTCC ¬= 0 THEN WRITE FTINCL FAILED
  ISPEXEC FTCLOSE
  SET &DATA1 =
  ISPEXEC LMINIT DATAID(DATA1) DDNAME(ISPFILE) ENQ(EXCL)
  IF &LASTCC ¬= 0 THEN WRITE LMINIT FAILED
  ELSE +
    DO
      ISPEXEC EDIT DATAID(&DATA1)
      ISPEXEC LMFREE DATAID(&DATA1)
    END
END
```

INDEX

Interactive System Productivity Facility/
Program Development Facility
Version 2 Release 2 for MVS SC34-4023-1
Services

READER'S
COMMENT
FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. It will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

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

Note: ***Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.***

Possible topics for comments are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name and mailing address:

____
____
____

What is your occupation? ______________________________

Number of latest Technical Newsletter (if any) concerning this publication: ________________

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

Cut or Fold Along Line

SC34-4023-1

Reader's Comment Form

Fold and tape

Please Do Not Staple

Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department T45, Information Development
P. O. Box 60000
Cary, North Carolina 27511

Fold and tape

Please Do Not Staple

Fold and tape

Cut or Fold Along Line

ISPF/PDF for MVS Services Printed in U.S.A. SC34-4023-1