Screen Definition Facility II

SH19-8211-00

Administrator's Guide

Release 4

Screen Definition Facility II

SH19-8211-00

Administrator's Guide

Release 4

Note!

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

First Edition (March 1995)

This edition applies to Release 4 Modification Level 0 of Screen Definition Facility II MVS, Program Number 5665-366, and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

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

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

IBM Vienna Software Development Laboratory
SDF Development
Department 00/174
Lassallestrasse 1
A-1020 Vienna
Austria

If you prefer to send comments by Fax, use this number:

Your International Access Code + 43 + 1 + 21145-4490

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

Contents

Figures

Notices

References in this publication to IBM* products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Commercial Relations, IBM Corporation, Purchase, NY 10577.

Programming interface information

This book documents General-use Programming Interface and Associated Guidance Information provided by SDF II MVS.

General-use Programming Interfaces allow to write programs that obtain the services of SDF II MVS.

General-use Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

General-use programming interface

General-use Programming Interface and Associated Guidance Information ...

End of General-use programming interface

Trademarks and service marks

The following terms, denoted by an asterisk (*), used in this publication, are trademarks or service marks of IBM Corporation in the United States or other countries:

BookManager	Common User Access	CUA
CICS/ESA	CICS OS/2	CICS/MVS
CICS/VSE	CICS/VM	GDDM
IBM	IMS/ESA	MVS/ESA
MVS/SP	MVS/XA	System /370
S/390	VisualGen	VTAM

About this book

This book is written for systems programmers, librarians, system administrators — people that will set up Screen Definition Facility II MVS (SDF II) in a software development environment. The book combines information that was previously kept in several books, and thus replaces:

- SDF II Installation and Migration Guide for MVS
- SDF II Diagnosis Guide

It now contains a new appendix that discusses the use of SDF II with ISPF/PDF SCLM.

The complete information on installing SDF II is now kept in one place, that is, in the *SDF II Program Directory* that comes with the machine-readable material.

New sections are:

- Adapting the SDF II installation
- Setting up libraries
- Migrating from one SDF II release to the next
- The variable pool
- Diagnosing error situations

Organization of this book

The chapters and appendixes of this manual are:

Appendix B, “SDF II variables” on page 139
Summarizes the use of the variables in the variable pool.

Appendix C, “Diagnosing error situations” on page 147
Describes what you need to do when you encounter an error that you wish to communicate to IBM.

Appendix D, “Using SDF II with ISPF SCLM” on page 151
Shows you how you can integrate SDF II into an application development environment that uses ISPF/PDL SCLM for its library and software configuration management.

A glossary, a list of SDF II publications, a list of referenced publications, and the index follow the appendixes.

Note: To reduce confusion, the SDF II terminology was clarified: The term “to import” is now used to mean “moving target system objects into SDF II”; this was previously called “to migrate.” The term “to migrate” is now used in its traditional sense, that is “moving to a changed operating environment,” in this book to move from SDF II Release 3 to SDF II Release 4.

Highlighting conventions used in this book

Font	Convention
Bold	Names of panels, fields, and function keys; phrases or paragraphs containing important information.
`Monospace`	Data set names, text displayed in the sample programs, text or commands you are instructed to type.
Italics	Words that are used for emphasis; variables in command strings.

Summary of enhancements

Summary of enhancements in SDF II Release 4

The most important user interface improvements use ISPF 4.1 functionality. They are available with the new Graphical User Interface (GUI) used by SDF II for OS/2 workstations and with the host user interface.

Functional improvements

Improved import

Support was added to create repeat formats and arrays.

Copy objects with their related objects

When copying a composite screen object, copying of its related objects can be requested.

Data extraction and modification utility

The extraction utility provides a means to extract data from an SDF II panel. The modification utility copies an SDF II panel and accepts changes from a user exit.

Interface for editing objects from the ISPF or SCLM editor

An edit macro allows to invoke the SDF II panel editor directly from the ISPF or SCLM Edit entry panel.

File tailoring within user exits

ISPF file tailoring services can be used in SDF II user exits without any restrictions.

Enhanced library search order

New options for the library search order for objects related to a primary object can be selected in the Specify Libraries dialog.

Changed target system defaults when generating or converting

When generating an object from the List Objects panel without specifying a target system, the object's target system is used. If this is ALL, the Identify Object for Generation panel is displayed. When converting an object from the List Objects panel without specifying a target system, the Specify Conversion Utility Parameters panel is displayed.

Command to show CUA types

A new command, **cualist**, shows the available CUA types and associated attributes.

Adapt data structure and format names

If a source panel that is created, copied, or renamed contains a format that has the same name as the source panel, this format's name will be changed to the panel name of the target panel.

If a source panel that is created, copied, or renamed contains a data structure that has the same name as the source panel, this data structure's name will be changed to the panel name of the target panel.

Default ddname for DCF output of print utility
When an SDF II object is printed as input for DCF and no data set name is specified for the print output, SDF II will write the output to a predefined ddname.

Support of SCLM alternate project definitions
SCLM alternate project definitions can be used for SDF II object libraries.

Support of flexible data set naming for SCLM
This allows to associate any library name with a specific project, group, and type.

Specification of library names
This allows greater flexibility in naming screen object libraries.

PICIN/PICOUT/DECPOS for BMS
Picture values or decimal position values specified in the panel editor are generated not only in the data structure but also in the BMS macros.

SDF/CICS compatible data structures
On the Specify Generation Parameters panel, SDF/CICS-compatible generation of adjunct names for the 3270 base attribute can be selected.

Hyphen within name of related field
The hyphen character (-) is accepted as part of the name of related fields.

Performance improvements

List Objects
The performance of the List Objects display has been improved.

Generation
Repeated loading of identical information is now avoided.

User interface and usability enhancements

Action bars
All data entry panels and window panels provide an action bar with pull-down menu choices.

Cursor-sensitive field level help
Help for any entry field can be obtained by placing the cursor on the field and pressing the help key.

Keylists
For each panel an appropriate set of PF keys (keylist) is defined.

Use of CUA element types
SDF II uses CUA types for its panel elements (except for dynamic areas). This allows the user of SDF II to apply global changes to the appearance of SDF II panels.

Use of GUI element types
Checkboxes and point-and-shoot fields are introduced.

Pop-up help panels
All help panels are displayed as pop-up windows.

Scrollable panels
All panels can be scrolled forward and backward.

Extended code page support
Extended code page support allows panels, messages, and other data to be displayed correctly on terminals using any ISPF-supported code page.

Target system specific panels
Data entry and menu panels contain only fields applicable for the current target system.

Simplified installation and customization
Online dialogs now help in customizing the routine that invokes SDF II. Modifications are kept for future customizations.

Closing quote for data set names optional
A closing quote is not required after data set names and SDF II library names.

Development environments

SDF II R4 operates under the currently supported releases of the following system environments:

- MVS/ESA SP* Version 5 (5655-068, 5655-069)
- MVS/ESA SP* Version 4 (5695-047, 5695-048)
- MVS/SP* Version 3 (5685-001, 5685-002)
- MVS/SP* Version 2 (5740-XC6, 5665-291)

or subsequent releases. SDF II also requires ISPF Version 4 for MVS (5655-042) and TSO/E Version 2 (5685-025) or later.

Requirements for printing: If output prepared by the SDF II print utility contains DBCS fields, you can use the Print Services Facility MVS (5665-275).

To print the field outlining attributes, the MVS/SP KANJI Print Utility (5799-BWM) is also required.

The Document Composition Facility (DCF) input prepared by the SDF II print utility can be processed with DCF (5748-XX9) Release 3.1 or later.

Output for the terminal printer, prepared with the SDF II print utility, can be printed with the print utilities of GDDM Version 2 MVS (5665-365).

Features: For importing objects from and generating objects for IMS/ESA*-MFS or IMS/VS-MFS, the MFS feature of SDF II is needed.

For national language translation services the national language support (NLS) features can be ordered.

Requirements for importing: For importing objects from and generating objects for IMS/ESA-MFS or IMS/VS-MFS, the MFS feature of SDF II is needed, which can be ordered separately.

In addition, objects created with the following IBM licensed programs can be imported:

- SDF/CICS OS/VS Release 5.0 (5740-XYF)
- SDF/CICS VSE Release 5 (5746-XXT)
- SDF/CICS CMS Release 1.0 (5664-178)

Chapter 1. Introduction

Screen Definition Facility II (SDF II) is an interactive tool that helps you develop and maintain panels, panel groups, partition sets, AID tables, and control tables. These SDF II objects can later be used in applications developed with or using CICS/BMS, IMS/MFS, ISPF, GDDM*-IMD, or CSP/AD.

With SDF II you can interactively develop, on any display device supported by ISPF Version 4 Release 1, any object that can be defined by using:

- CICS/BMS macros
- IMS/MFS utility control statements
- ISPF panel definitions
- GDDM-IMD
- CSP/AD

To import objects into SDF II from IMS/MFS and to generate objects for IMS/MFS, you need the MFS feature of SDF II.

SDF II functions

The following SDF II dialogs are available to all SDF II users:

Panel editor

The main components of a panel are its format and the data structure. The format describes how data is to be presented on a particular device. The data structure describes how data is communicated from and to the application program. Other components of a panel are the panel characteristics and tables containing symbols with special meanings for the definition process. SDF II distinguishes between general panel characteristics and those specific to a target system.

Panel group editor

A panel group is an object containing information required by CICS/BMS, GDDM-IMD, or CSP/AD run-time services. The components of a panel group are its characteristics, a list of all panels to be contained, and, for GDDM-IMD, a list of all programmed symbol sets used explicitly or implicitly by the panel group.

Partition set editor

A partition set defines how the screen of an IBM 8775, 3290, or 3180 display device is split into a number of separately controlled areas. The components of a partition set are its characteristics and a list of all partitions contained in the partition set.

AID (attention identifier) table editor

An AID table maps operator actions to values, which are then returned to the application program.

Control table editor

A control table defines a sequence of conditional operations and their associated control or branching functions.

Generation

Panels, panel groups, and partition sets can be generated for use in the target systems as follows:

Target system	Generated objects
CICS/BMS	CICS/BMS macros, application data structures
IMS/MFS	IMS/MFS utility control statements, application data structures
ISPF	ISPF panels, data structures
GDDM-IMD	GDDM-IMD export data sets
CSP/AD	CSP/AD export data sets or CSP/AD external source format.

The generation of objects for IMS/MFS requires the optional MFS feature.

User exits are provided to invoke an EXEC after generation, to further process the generated output.

List objects

The List Objects panel gives the names and types of all objects that satisfy the generic search argument. It also allows the user to perform actions on these objects, such as invoking the editors, generating objects, or printing.

Specify libraries

In this dialog, the users specify the libraries to be accessed and the search order. These libraries are then available for all SDF II dialogs. They are accessed through ISPF. The libraries can be controlled by Software Configuration and Library Manager (SCLM) or any other library manager. Users can take advantage of the functions of the ISPF Library Management Facility.

Utilities

- With the *print utility*, users can print a representation of any SDF II object. The user can select whether the output is for the system printer, a DBCS printer, or the GDDM print utility, or whether it is to be used as input to the Document Composition Facility (DCF).
- With the *import utility*, users can import the following into an SDF II supported library:
  - Maps, map sets, and partition sets defined with CICS/BMS macros.
  - Format sets defined with IMS/MFS utility control statements. (To import IMS/MFS objects you need the optional MFS feature.)
  - Panels defined in the panel syntax of ISPF Version 2 or later.
  - Maps, map groups, and AID tables defined by and exported from GDDM-IMD.
  - Maps and map groups defined by CSP/AD external source format or by extended external source format.
  - Maps and map groups defined by and exported from CSP/AD.
  - Maps, map sets, and partition sets defined and dumped with SDF/CICS.

- With the *conversion utility*, users can convert SDF II objects defined for one target system to a format suitable for the use in another target system.
- With the *print online reference utility*, users can print all or part of the online reference.
- With the *panel construction utility*, users can construct a panel from a list of panel elements.

Profile

Users can define defaults that are used in SDF II dialogs to tailor the environment to their specific needs with the profile editor.

Extraction Utility

This utility allows users to extract information about a panel and its fields. such as the field's position, its length, or its contents. This information is passed to a user exit where any further processing can be done.

Modification Utility

This utility is similar to the extraction utility but offers an extra function in the user exit that allows to modify the field contents. This utility can also copy formats and change the device of a panel.

SDF II application prototype

The SDF II application prototype function allows you to review the flow of panels, together with the eventual user of the application, before anyone writes a line of code. This helps you obtain early feedback that can be incorporated into the design of the application without anyone having to rewrite the application program code.

Generally, there are two types of application prototypes:

- A simulative prototype, which is used early in the development cycle. It consists of a series of panels, representing a first approximation of the application's panels.
- An operational prototype, which is a limited function version of the application program. It can be as simple or complex as necessary.

SDF II supports both simulative and operational prototypes. It provides all the functions needed to define and run a prototype. By adding your own routines, you can make the prototype as complex as you want, thereby moving towards an operational prototype.

Online reference

The online reference contains all the information you need for working with SDF II. How to use it, and how to obtain a printed copy is discussed in *Designing Panels with SDF II*, SH19-8212-0.

SDF II customization dialogs

The following SDF II dialogs are intended for the system programmer and the national language translator:

- Device type editor

  The system programmer can define the characteristics of devices referred to in the panel, panel group, and partition set editor dialogs, and group devices to device lists, with the device type editor.

- National language translation support

  The contents of the windows of the SDF II panels, the panel commands, the line commands, and the operator input can be translated in dialogs provided in SDF II.

  SDF II messages and tutorial topics, which are contained in ISPF message and panel libraries, can be translated using any standard text editor.

- Emphasis class table definition

  Attributes can be grouped into emphasis classes. In the panel editor, users can assign emphasis classes instead of assigning single attributes.

- CUA* attribute definition

  The attributes of CUA panel element types can be changed.

Chapter 2. Adapting the SDF II installation

Adapting the invocation environment

The routines to call SDF II are DGIIXSDF and DGIIXSFC. SMP/E defines the following aliases:

SDF2 For all definition, generation, utility dialogs, and the prototype facility

SDF2C For the customization dialogs: the device table editor, the translation dialogs, the emphasis class table definition, and the CUA attribute definitions. This routine should be accessible only by the system programmer.

Both routines call the invocation routine DGIIXINV, which is generated by DGIIXIN1 and has the alias SDF2INV. It contains the necessary TSO ALLOC statements and ISPF LIBDEF statements for the ISPF and the SDF II libraries. SDF2, SDF2C, and SDF2INV are REXX EXECs.

Generating the invocation routine

All adaptations are done by routine DGIIXIN1, which displays panels and prompts for all definitions. When leaving this routine, a customized DGIIXINV routine is generated. All user input is saved and can be used for future customization.

The customized DGIIXINV routine retains all modifications and will be used as input for future customization and for migration to higher SDF II levels.

In front of each ISPF library for which SDF II allocations are made (ISPPLIB, ISPTLIB, ISPSLIB, ISPMLIB, ISPLLIB) a private library may be allocated in which customized SDF II objects can be kept.

The SDF II invocation routine calls a user exit that allows further customization (see “Optional user exit” on page 12)

Note: All SDF II libraries are allocated according to the specification of the installation's library prefix. You need to specify libraries only if you need additional allocations for private libraries.

When ISPF is started, invoke this program as follows:

```
tso ex 'SDF2.V1R4M0.SDGICMD(DGIIXIN1)'
```

Note: Before running DGIIXIN1, check that it has the correct definition of panel libraries, for instance:

```
/* ---------------------------------------------------------------- */
/* NOTE 1:                                                          */
/*    Before running this procedure check or correct DGIIIPFX below */
/*    to the data set prefix of installed SDF II datasets           */
/* ---------------------------------------------------------------- */
  DGIIIPFX = "SDF2.V1R4M0"
```

General definitions

```
DGIIE800
COMMAND ===> _____________________________________________________________
                        Define SDF II Invocation Defaults
                     DGIIXINV options for Interactive and Batch
1
Comment  . . . Customized DGIIXINV_____________________________________
2                                                                 More:     +
3       Old DGIIXINV   PRIVATE.CLIST(DGIIXINV)_____________________________
        New DGIIXINV   PRIVATE.CLIST(DGIIXINV)_____________________________
4
SDF II Prefix  SDF2.V1R4M0._______ SDF II Default Language  EU

                          Private SDF II Library Definitions
5
6       Private Tutorial Library . . .  ___________________________________
7       Private Panel Library  . . . .  ___________________________________
8       Private Message Library  . . .  ___________________________________
9       Private Table Library  . . . .  ISPF.ISPPREU_______________________
10      Private Table Output . . . . .  ISPF.ISPPREU_______________________
11      Private Skeleton Library . . .  ___________________________________
        Private Load Library . . . . .  ___________________________________

                             Printer Output Definition
12
13      SDF II Print and Trace Output . SYSOUT(A)_________________________
14      SDF II DBCS Print . . . . . . . ___________________________________
        SDF II DBCS Print Outlined. . . ___________________________________

              PF3 to create new EXEC or enter  CANCEL  to terminate
```

1. The comment that will be copied to the output DGIIXINV for documentation.
2. The name of an existing DGIIXINV that can be used to copy old customization information into the new procedure.

   Note: The old DGIIXINV must have been generated by DGIIXIN1 procedure.
3. Name of the new DGIIXINV procedure
4. Prefix of SDF II libraries in your installation. The default language may be EU, DU, DS, JN, or EP.
5. Private tutorial library. If specified here it will be allocated before the SDF II tutorial library.
6. Private panel library. If specified here it will be allocated before the SDF II panel libraries.
7. Private message library. If specified here it will be allocated before the SDF II message library.

   Note: Use the same name as under item 9.
8. Private table library. If specified here it will be allocated before the SDF II table library.
9. Private table output library. If not specified, `ISPF.ISPPR`*`langcode`* will be used.
10. Private skeleton library. If specified here it will be allocated before the SDF II skeleton library.

11. Private load library. If specified here it will be allocated before the SDF II load library.
12. SDF II print and trace output definition.
13. SDF II DBCS Print output definition.
14. SDF II DBCS Print outlined output definition.

Notes:

1. Define your private panel, table, message, and skeleton libraries if you make private modifications to these libraries. They can be easily saved for any future SDF II migrations.
2. Make all library definitions according to TSO naming conventions, use apostrophes (') for a fully qualified name and use no apostrophe if the data set name is to be prefixed by the user ID. If more libraries are specified, separate them with a blank or a comma.
3. You may use the term `USERID()` in a data set name definition:

   ```
   DGI Object Set 1  'TEST.USERID()'
   ```

   During invocation of DGIIXINV it will be substituted with the current user ID.

Define SDF II object libraries

This defines the initial and default setting of the object libraries for the first invocation of SDF II.

```
DGIIE800
COMMAND ===> _____________________________________________________________
                    Define SDF II Invocation Defaults
                  DGIIXINV options for Interactive and Batch

Comment  . . . Customized DGIIXINV_____________________________________________
                                                                More:   - +
                            SDF II Object Libraries

                   DSname-Prefix                     Description

1
1       SDF II Object Set 1 _____________________________ ____________________
1       SDF II Object Set 2 _____________________________ ____________________
1       SDF II Object Set 3 _____________________________ ____________________
1       SDF II Object Set 4 _____________________________ ____________________
1       SDF II Object Set 5 _____________________________ ____________________
1       SDF II Object Set 6 _____________________________ ____________________
1       SDF II Object Set 7 _____________________________ ____________________
1       SDF II Object Set 8 _____________________________ ____________________
2       SDF II Object Set 9 _____________________________ ____________________
        SDF II Autosave Lib: ____________________________

           PF3 to create new EXEC or enter  CANCEL  to terminate
```

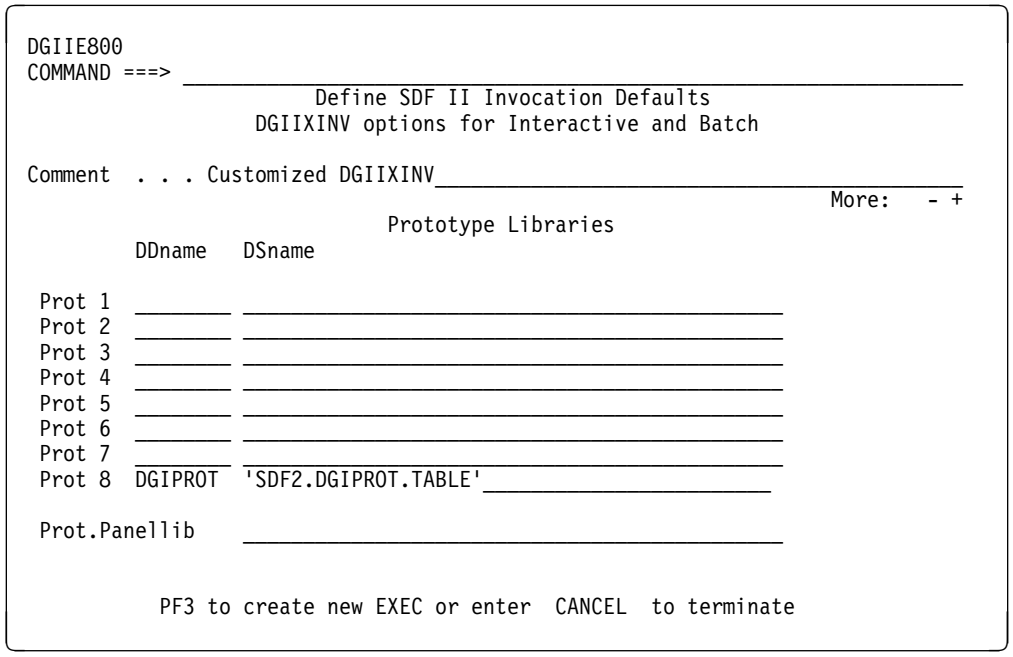
1. Define up to nine SDF II object libraries and add a short description.

   Note: The last qualifier of the data set name need not be specified. It is added by SDF II depending on the type of the object (DGIPNL, DGIGRP, DGIPST, DGIOCT, or DGITBL).

2. Data set name of SDF II autosave library

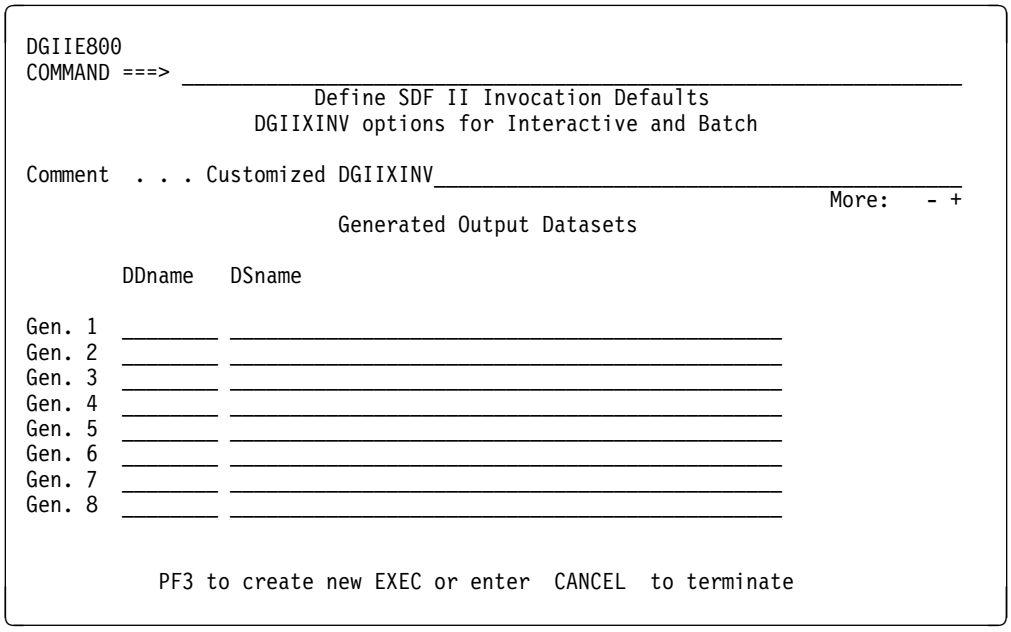
Note: These definitions will be overwritten when you define your object libraries online (using the Define Libraries panel).

Define prototype libraries



1. Define the ddname and data set name for prototype data sets.
2. Define the data set name for the prototyping panel input library.

Define generated output data sets



1. Define ddname and data set name for generated output data sets.

Optional definitions and allocations

```
DGIIE800
COMMAND ===> ____________________________________________________________
                    Define SDF II Invocation Defaults
                 DGIIXINV options for Interactive and Batch

Comment  . . . Customized DGIIXINV_____________________________________
                                                              More:   -
                          Optional Invocation Exit
1
Users Exit to DGIIXINV  ________


                            Optional Allocations
2
Need SCLM-Allocation ?   NO_
3
Need ISPF-Allocation ?   NO_
4
5       ISPF Level       4.1        ISPF Language   EU
6       ISP Prefix       ISP.V4R1M0._____________________________
7       ISR Prefix       ISR.V3R5M0._____________________________
        Profile Dataset  ISPF.ISPPREU_____________________________

           PF3 to create new EXEC or enter  CANCEL  to terminate
```

1. Define the name of a user exit that allows to make additional changes during invocation. (See “Optional user exit” on page 12.)
2. Need allocation for SCLM data sets DGILX01, DGILX02, and DGILXOL. When specifying YES also check user exit DGILXOL.
3. If YES is specified, DGIIXINV allocates the required ISPF data sets when invoked from batch. If NO is specified, you have to provide the correct allocations for ISPF in your job control. If NO is specified, items 2 to 5 are ignored.
4. Specify the level and language of the installed ISPF system. Language may be EU, DU, DS, JN, or EP.
5. Specify the prefix for ISP data sets.
6. Specify the prefix for ISR data sets (if level 3.5 or newer).
7. Specify the name of the ISPF profile data set.

Optional user exit

If a user exit is defined (see “Define prototype libraries” on page 9, item 5) it allows additional user modifications.

When this exit is called, the ISPF application is set to DGI or DGIS when invocation is for SCLM. All variable names listed in section “Variable names available in user invocation exit” on page 13 are available in the ISPF variable pool and can be modified. After modification, the variables must be moved back into the ISPF variable pool. In addition, all variables in the DGI profile can be set to modify SDF II run time characteristics.

On entry to this exit, one argument is passed to indicate the status of SDF II:

BEGIN The exit is called before SDF II invocation and before allocating data sets.

END The exit is called after SDF II invocation. The variable DGIIIIRC contains the SDF II return code.

A sample user exit routine (DGIIXIN2) is supplied.

```
/* --- REXX ------------------------------------------------------- */
/* This procedure is called from customized DGIIXINV procedure       */
/* to allow additional user modification of invoke-variables or      */
/* to set ISPF variables for later use in SDF II processing.         */
/*                                                                   */
/* There is one argument passed to this exit:                        */
/*                                                                   */
/*   'BEGIN' exit is called before SDF II is scheduled               */
/*                                                                   */
/*   'END'   exit is called after  SDF II has finished               */
/*                                                                   */
/* On entry ISPF environment is set to NEWAPPL(DGI).                 */
/* Invoke-variables are available in ISPF variable pool.             */
/* ----------------------------------------------------------------- */
/* User's code follows here:                                         */
/* ----------------------------------------------------------------- */
   address ISPEXEC 'CONTROL ERRORS RETURN'
   select
     when arg(1)='BEGIN' then do
/*                                  modify language code             */

       if date('w')='Tuesday' then dgiiilng='EU'
                              else dgiiilng='DU'
       address ISPEXEC 'VPUT (DGIIILNG)'
       end
     when arg(1)='END' then do
       end
     otherwise nop
     end
/* ----------------------------------------------------------------- */
                                                          return
/* ----------------------------------------------------------------- */
```

Variable names available in user invocation exit

These variables contain the values set by previous actions on the panels.

Figure 1. User invocation exit - variables

Name	Description
DGIIIPFX	SDF II Prefix of installed libraries
DGIIILNG	SDF II Language Code
DGIIIULB	Private Library: Tutorial Library (DGIULIB)
DGIIIPLB	Private Library: Panel Library (ISPPLIB)
DGIIIMLB	Private Library: Message Library (ISPMLIB)
DGIIITLB	Private Library: Table Library (ISPTLIB)
DGIIITBL	Private Library: Table Output (ISPTABL)
DGIIISLB	Private Library: Skeleton Library (ISPSLIB)
DGIIILLB	Private Library: Load Library (ISPLLIB)
DGIIIPRX	Print Output Definition (DGIPRINT)
DGIIIDBC	Print Output for DBCS (DGODBCS)
DGIIIDBO	Print Output for DBCS Outlined (DGIDBCSO)
DGIIIOLn	SDF II Object Library 1 ... 9 (ILV10L01 ... ILV10L09)
DGIIIODn	SDF II Object Library 1 ... 9 Description (ILV10D01 ... ILV10D09)
DGIIIOLA	SDF II Object Autosave Library (ILV10L10)
DGIIIPFn	Prototype Library ddname 1 ... 8
DGIIIPDn	Prototype Library data set name 1 ... 8
DGIIIPPU	Prototype Panel Input data set name (ISPPUSR)
DGIIIGFn	Generated Output: ddname 1 ... 8
DGIIIGDn	Generated Output: data set name 1 ... 8
DGIIISCL	Need SCLM allocations (Yes or No)
DGIIIREQ	SDF II Request
DGIIITRC	Trace Option (OFF|ON|ALL)
DGIIIIRC	SDF II return code

What DGIIXINV is doing

SDF II is invoked via the DGIIXINV routine, which sets up the SDF II environment. Before calling the SDF II program, the following is done.

For batch invocation, if requested via DGIIXINV customization (done by DGIIXIN1):

```
ALLOC ISPPROF
ALLOC ISPLLIB
ALLOC ISPMLIB
ALLOC ISPTLIB
ALLOC ISPSLIB

ALLOC ISPPALT       if language JN
ALLOC ISPMALT       if language JN
ALLOC ISSMALT       if language JN
```

For SDF II invocation, add SDF II libraries via LIBDEF and make other SDF II libraries available via ALLOC:

```
LIBDEF ISPLLIB      SDF II load library
LIBDEF ISPPLIB      SDF II panel library
LIBDEF ISPMLIB      SDF II message library
LIBDEF ISPTLIB      SDF II table library
LIBDEF ISPTABL      SDF II table output library

LIBDEF ISPSLIB      SDF II skeleton library
```

Allocation for all defined SDF II prototype libraries (SHR)
Allocation of ISPPUSR for prototype panel input (SHR)

Allocation of all defined SDF II generation output libraries (OLD)

```
VPUT                          Prefix of all SDF II object libraries.

ALLOC DGIPRINT if defined     Print and trace output
ALLOC DGIDBCS  if defined     DBCS output file (OLD)
ALLOC DGIDBCSO if defined     DBCS outlined output file (OLD)

ALLOC DGIULIB  if defined     SDF II online reference (SHR)
```

Making the invocation routine accessible

To be able to access the invocation REXX EXECs, do either of the following:

- Allocate SDF2.V1R4M0.SDGICMD to SYSEXEC in your logon procedure.
- Copy the EXECs to a data set allocated to SYSPROC.

For more information refer to *TSO/E Version 2 REXX Reference*.

Making ISPLINK accessible

Make sure that ISPLINK is in a system library that is in the search sequence for an MVS load. This is necessary because ISPLINK is not linked into SDF II load modules, but is dynamically loaded during execution.

Definition of the ISPF libraries and the ISPF profile

Check the assignment statements that define the ISPF libraries and the ISPF profile when generating the invocation routine SDF2INV. See "Optional definitions and allocations" on page 11. If you use different library names or different parameters for the ISPF profile, change these statements accordingly. The generation routine DGIIXIN1 will generate correct ISPF data set names depending on the ISPF version you specify.

Note: The definition of the ISPF profile is different from the definition in SDF II Release 2. The new names use the same two-letter language suffixes as the other data sets. If you have SDF II Release 2 installed, either change the statements during generation of SDF2INV to make them match the names you use, or rename the profiles of all users.

Destination of the print output

In the SDF II print utility (option **9.1** of the **sdf2** command), you can specify the destination of the SDF II output. The output can be prepared for:

1. The standard printer
2. The GDDM-IMD print utility
3. Input to DCF
4. A DBCS printer
5. A DBCS printer with field outlining

Note: The SDF II print utility uses ISPF or ISPF temporary data sets. These data sets are dynamically allocated during the print operation. However, it is recommended to preallocate these data sets to VIO, to avoid performance problems. Refer to the appropriate ISPF or ISPF installation guide for information on how to preallocate temporary data sets.

Output for the standard printer: Output for the standard printer is stored in the data set allocated to the ddname DGIPRINT.

When a preallocated data set for DGIPRINT is to be used, it must be allocated with record format FBA and a record length of 121. During definition of input for generation of the SDF2INV routine, specify the name of this data set instead of SYSOUT(A).

Output for the GDDM print utility: If you want to send output to the GDDM print utility, add the GDDM program library to the libraries to be allocated to ISPLLIB. See "General definitions" on page 6, item 11.

When you use the SDF II print utility later, you need to provide the user with the printer identification. The printer identification is the device ID defined in SYS1.VTAMLIST for TSO under VTAM.

Refer to *GDDM Installation and System Management* for information on how GDDM processes the output.

Output used as input to DCF: Output from SDF II that is to be used as input to DCF is stored in the partitioned data set you defined on the Specify Print Utility Parameters panel as member *objectname*, where *objectname* is the name of the object to be printed. This data set needs to be allocated with record format V and a record length equal to the print width defined with the SDF II profile editor. The default print width is 120 bytes.

To allocate this data set, you can use the DGIIXLIB EXEC from the SDGISAM library.

Output for the DBCS printer: To print output on the DBCS printer, the Print Services Facility (PSF) must be installed.

Output for the DBCS printer is stored in the data set allocated to the ddname DGIDBCS. It needs to be allocated with record format V and a record length of 319.

All print output from one session is stored in one member.

To allocate this data set, you can use the DGIIXLIB EXEC from the SDGISAM library. To use the data set, define its name during definition of input for generation of the SDF2INV routine. See “General definitions” on page 6, item 13.

Output for the DBCS printer with field outlining: To print output on the DBCS printer, the Print Services Facility (PSF) must be installed. Process the output with the DBCS print utility.

Output for the DBCS printer is stored in the data set allocated to the ddname DGIDBCSO. It needs to be allocated with record format V and a record length of 637.

All print output from one session is stored in one member.

To allocate this data set, you can use the DGIIXLIB EXEC from the SDGISAM library. To use the data set, define its name during definition of input for generation of the SDF2INV routine. See “General definitions” on page 6, item 14.

Defining prototype libraries

To be able to use the prototype facility, define the name of the library that is to contain the output of the generation utility in definition of input for generation of the SDF2INV routine. (see “Define prototype libraries” on page 9, item 2). This ensures that this library is defined as ISPF panel input library.

You may also modify the definition for DGIPROT. This ISPF table library is used in the prototype facility to store the prototype tables. If you want to use other prototype libraries, define them during definition of input for generation of the SDF2INV routine. See “Define prototype libraries” on page 9, item 1.

Generated output

If you want to specify ddnames for the generated output, define them during definition of input for generation of the SDF2INV routine. See “Define prototype libraries” on page 9, item 3. The data sets are allocated with disposition OLD, but note that this gives you exclusive access to the data sets.

Allocation Summary

Figure 2 summarizes allocation requirements and shows the default values used by the DGIIXLIB EXEC in the SDGISAM library.

Figure 2. Allocation requirements

Data Set	DSORG	RECFM	LRECL	BLKSZ	PRI	SEC	DIR
Output DCF format	PO	VB	120	6120	1000	200	100
Print Output DBCS	PS	VB	319	6061	1000	200	-
Print Output DBCS Outl.	PS	VB	637	5733	1000	200	-
Prototype Libraries	PO	FB	80	6160	200	50	50
Generated Output Library	PO	FB	80	6160	1000	200	100
SDF II Object Libr. DGIPNL	PO	FB	80	6160	1000	200	100
SDF II Object Libr. DGIGRP	PO	FB	80	6160	100	50	50
SDF II Object Libr. DGIPST	PO	FB	80	6160	100	50	50
SDF II Object Libr. DGITBL	PO	FB	80	6160	100	50	50
SDF II Object Libr. DGIOCT	PO	FB	80	6160	100	50	50
(SDF II DGIPRINT Dataset	PS	FBA	121	6050	100	50	-)

Note: Primary and secondary quantities are in blocks.

Chapter 3. Setting up libraries

This chapter contains General-Use Programming interface and Associated Guidance information.

This chapter describes the types of libraries used by SDF II and how to set up these libraries. For controlled libraries, it also describes how to adapt the user exit routines that are needed for using these libraries.

Types of objects

Objects used with SDF II are:

- SDF II objects
- Generated data structures
- Generated control block source
- Imported input
- Prototypes
- Listings

SDF II objects

These are objects produced by an SDF II editor. They are internal to SDF II and should be modified only by SDF II. SDF II objects are panels, panel groups, partition sets, AID tables, and control tables.

Figure 3 lists the five types of SDF II objects and shows which object can be produced for which target system.

Figure 3. SDF II object types

Object	Type	IMS/MFS	CICS/BMS	GDDM-IMD	CSP/AD	ISPF
Panel	DGIPNL	√	√	√	√	√
Panel group	DGIGRP		√	√	√	
Partition set	DGIPST	√	√			
AID table	DGITBL	√		√		
Control table	DGIOCT	√				

Generated data structures

These are data structures produced by the generation function of SDF II to be included in application programs that run under the CICS/BMS, IMS/MFS, or ISPF target system.

Generated control block source

These are objects produced by the generation function of SDF II for use in one of the SDF II target systems. Generated control block source is CICS/BMS macros, IMS/MFS utility control statements, CSP/AD external source format or export data sets, ISPF panels, or GDDM-IMD export data sets.

Figure 4 shows which control block source can be generated from which SDF II object and for which target system.

Figure 4. Generation of control block source

SDF II Object	IMS/MFS	CICS/BMS	GDDM-IMD	CSP/AD	ISPF
Panel	Format set	Map	Export data set	External source format or export data set	Panel
Panel group		Map set	Export data set	External source format or export data set	
Partition set		Partition set			

Import data
This is control block source or SDF/CICS dump data sets created outside of SDF II to be imported into SDF II. Import data can be CICS/BMS macros, IMS/MFS utility control statements, CSP/AD external source format or export data sets, extended external source format, ISPF panels, GDDM-IMD export data sets, or SDF/CICS dump data sets.

Prototypes
These are objects produced by the SDF II prototype facility to be used for prototype simulation in SDF II. Prototypes are ISPF tables.

Listings
Contain output from the print utilities and messages from utilities, such as the generation or import utility. See also “Destination of the print output” on page 15.

Types of libraries and data sets

Libraries for the objects used with SDF II must be allocated and defined before SDF II is invoked. These libraries can be:

- ISPF libraries
- SCLM-controlled libraries
- Externally controlled libraries
- Partitioned data sets
- ISPF table libraries
- A sequential data set

The following lists the libraries that can be used for each object type:

Libraries for SDF II objects

- ISPF libraries
- SCLM-controlled libraries
- Externally controlled libraries

Controlled libraries may not be used as autosave libraries.

The libraries must be allocated. On the Specify Libraries panel, the library names must be specified and a library ID assigned to each library.

The library name is as follows:

- For externally controlled libraries, a name determined by the system programmer and specified in the user exit routine DGILXLI.

The library ID is a number that represents:

- For ISPF libraries, one group of up to five ISPF libraries, each of which is used to store objects of one SDF II object type (as listed in Figure 3 on page 19).
- For ISPF SCLM libraries, one path in the SCLM hierarchy. The path is identified by the project name and the group name of the user level that the system programmer has specified in the user exit routine DGILXLI.
- For externally controlled libraries, what was determined by the system programmer and specified in the user exit routine DGILXLI where the internal organization of the library is defined.

For more information about specifying libraries refer to *SDF II General Introduction*, chapter *Setting up your SDF II session*, under *Specifying libraries*.

Libraries for generated data structures

- Partitioned data sets
- SCLM-controlled libraries

These libraries need to be specified on the Specify Generation Parameters panel and for SCLM libraries in the request file.

Libraries for generated control block source

- Partitioned data sets
- SCLM-controlled libraries

These libraries need to be specified on the Specify Generation Parameters panel and for SCLM libraries in the request file.

Libraries for imported input

Imported input is stored in partitioned data sets. They need to be specified on the Specify Import Parameters panel.

Libraries for prototypes

Prototypes are stored in ISPF table libraries. These libraries are specified in the SDF2INV EXEC. See “Defining prototype libraries” on page 16.

Data sets for listings

For listings, a sequential data set is specified in the SDF2INV EXEC by setting the variable DGIPRINT to the parameters for the ALLOC statement. See “Destination of the print output” on page 15.

For SCLM, the library needs to be specified in the SCLM project definition.

Defining ISPF/PDF libraries

To define ISPF libraries for SDF II objects, follow the procedure outlined in the following steps.

Step 1: Identifying required libraries

First use Figure 3 on page 19 to identify the types of SDF II objects that are to be used in your installation. These types identify the number of libraries necessary per library group.

Step 2: Defining the ISPF libraries

Use the ISPF data set utility (option **3.2**) to define your ISPF libraries.

Allocate the data sets with record format FB and a record length of 80.

For each library group, define one library for each object type to be used. The syntax of the library names is that of an MVS data set name with any number of qualifiers, where SDF II appends the lowest qualifier (see Figure 3 on page 19, column **Type**).

Step 3 (optional): Adapting user exit routine DGILXLI

The shipped version of DGILXLI returns with a return code of 0.

To limit the users' access to certain libraries, you can specify these libraries in DGILXLI. Otherwise, you do not need to customize the DGILXLI EXEC.

To customize the EXEC, take the member DGILXLI from SDF2.V1R4M0.SDGISAM as an example and proceed as follows:

1. Find these comment lines:

   ```
   /*********************************************************************
   * Definition of supported libraries
   *
   ```

2. After the comment box, add lines for each library to be made accessible, such as:

   ```
   libs=n                          /* The library counter            */
   name.x="libraryname"            /* The library name               */
   type.x="PDF"                    /* The library type is PDF        */
   ```

   Where:

   n is the number of library specifications that follow.

   x is 1 for the first library, 2 for the second, and so forth.

   libraryname is the name of the library that is to be used on the Specify Libraries panel; it must be exactly as entered on the Specify Libraries panel, including any quotes if present.

3. Find these comment lines:

   ```
   /*-------------------------------------------------------------
   * Check the passed library name
   *-------------------------------------------------------------*/
   ```

4. After the comment box, change the next `Exit 0` statement that follows this comment to:

   ```
       Exit 8                       /* A PDF library               */
   ```

   This return code will indicate an invalid library specification (see Figure 5 on page 29).

Defining SCLM-controlled libraries

When SCLM libraries are used to store SDF II objects, the processing is as follows:

- The users edit their objects in SDF II. If an object has already been promoted to a higher level in SCLM, SDF II draws it down to the user level and locks it for other users. It remains locked until the user builds and promotes it again.

  To allow editing of SDF II objects from an SCLM environment, SDF II can be called from an SCLM edit macro via the invocation interface. If you need to set the target system for an edit request, add a column DGIRVTS to the invocation interface table and set it to any of the following values:

  C For target system CICS/BMS
  M For target system IMS/MFS
  I For target system ISPF
  G For target system GDDM-IMD
  X For target system CSP/AD
  * For target system ALL

- To perform the BUILD, SCLM services are used. SCLM internally calls SDF II to generate all involved objects as entered in the architecture definition file.
- To promote objects to a higher level, also SCLM is used. The object will be unlocked by SCLM when it is promoted from the user level to a higher level.

Objects stored in an SCLM controlled library can be:

- SDF II objects
- Generated data structures
- Generated control block source
- Listings

SCLM libraries are partitioned data sets.

See *ISPF Software Configuration and Library Manager (SCLM) Project Manager's Guide* for details.

Prerequisites

It is assumed that an SCLM project definition already exists. For each SCLM project definition, determine:

- The target system to be used in your installation.

  In one project definition only one target system should be used.

- The supported programming languages for which data structures will be generated.
- For the CICS/BMS target system, the device types for which objects will be generated.

  SDF II generates one output per specified device type. No device lists may be used to define panels or panel groups.

- The object types to be generated.

  Per target system, decide according to Figure 4 on page 19, which SDF II object types you want to generate (BUILD).

Either panels or panel groups should be generated. For the CICS/BMS target system, also partition sets can be generated.

Step 1: Identifying required libraries

As a first step, identify the types of objects you want to store and select the appropriate SCLM object type.

- SDF II objects

  According to your target system and the types of objects you intend to produce, select the appropriate names from column **Type** in Figure 3 on page 19.

- Generated data structures

  For the data structures to be generated, you need one SCLM object type per programming language for which you want to generate data structures. You can choose the names for these object types freely.

- Generated control block source

  For the CICS/BMS target system, panel groups or panels and partition sets can be generated. In each case, one output data set is generated per specified device type. You need one SCLM object type for each generated output data set.

  For any other target system, you need one SCLM object type. You can choose the names for these types freely.

- Listings

  For listings, you need one SCLM object type for each SDF II object type for which you want to perform a BUILD. You can choose the names for these object types freely.

Step 2: Allocating partitioned data sets for SCLM

Allocate partitioned data sets for the SCLM object types that you determined in the previous step. For listings, use record format FBA and a record length of 121. For export data sets (GDDM-IMD or CSP/AD) use record format FB and a record length of 256. In all other cases use record format FB and a record length of 80.

Use the ISPF data set utility (option **3.2**) to define your ISPF libraries.

Alternatively, you can use the DGIIXLIB EXEC from SDF2.V1R4M0.SDGISAM, as described in the *SDF II Program Directory* that comes with the SDF II distribution tape.

Step 3: Adapting the SCLM project definition

You need to adapt the SCLM project definition source file to specify the necessary object types and to add the required languages.

Define object types

To your SCLM project definition files, add one `FLMTYPE` macro for each SCLM object type to be defined.

Define required languages for SCLM

Depending on your target system, copy and adapt file DGIILAN*n* (where *n* is 1 for CICS/BMS, 2 for IMS/MFS, 3 for ISPF, 4 for GDDM-IMD, or 5 for CSP/AD) from SDF2.V1R4M0.SDGISAM into the macro source library of your project (*projid*.PROJDEFS.SOURCE). Add the required languages to your project definition file. One set of language definition is necessary for each SDF II object type you want to create in your installation.

SCLM language definition macros: To produce the language definitions for SCLM, use the following macros and appropriate parameters.

FLMLANGL macro

This is the SCLM language definition macro. You need one for each SDF II object type you want to create.

LANG={DGIGRP|DGIPNL|DGIPST|DGIOCT|DGITBL}

Specifies the language. The language must be identical to the SDF II object type name.

VERSION=V1R4M0,

Identifies the current SDF II version, release, and modification level.

FLMTRNSL macro

This macro defines the translators. You need one translator for each language (that is, each SDF II object type) for which you want to perform a build.

CALLNAM='SDF2 generation'

Specifies the name of the translator. It can be chosen freely. The name assigned here is only a suggestion.

FUNCTN=BUILD

Identifies the function performed by the translator.

COMPILE=IRXJCL

Identifies the interface routine.

CALLMETH=LINK

Gives the translator access to ISPF variables.

PORDER=1

Indicates that the options specified in the OPTIONS parameter below are passed.

VERSION=V1R4M0

Identifies the current SDF II version, release, and modification level.

OPTIONS='... '

Specifies the options passed to the BUILD routine IRXJCL. The options consist of 2 parts separated by a slash (/):

- The first part consists of:
  1. DGIIXISP - the Interface routine to ISPF called by routine IRXJCL. This EXEC is shipped with SDF II and performs:

     ```
     Address ISPEXEC "SELECT CMD("ARG(1)")"
     ```

  2. DGIIXBLD - the REXX EXEC called by ISPF. This is the translator interface to SDF II. This EXEC is shipped with SDF II.
  3. @@FLMINC - SCLM replaces this variable by the address of an area where SDF II puts the name and type of related (included) objects. (Dynamic Include Tracking.)
  4. The name of the request file that is to be used by SDF II. See “Step 4: Preparing build request files” on page 27.

- The second part contains data that can be used in the request file via the variable $args. The data items, which are listed below, can be accessed using the REXX built-in function WORD in the form WORD($args,*n*).
  1. @@FLMGRP - SCLM passes the name of the group of the level for which the BUILD has to be performed.
  2. @@FLMTYP - SCLM passes the name of the object type.
  3. @@FLMMBR - SCLM passes the name of the member.
  4. The library name as entered on the Specify Libraries panel (and used in DGILXLI). If the project name has been chosen to be identical to the library name (see name.*x* in "Step 5: Adapting user exit routine DGILXLI" on page 27), you can provide the SCLM variable @@FLMPRJ.
  5. The ddnames used for generation output.

It is also possible to pass additional items to the request file.

FLMALLOC macro

This macro defines the listing.
You need one for each BUILD translator.

IOTYPE=O
Identifies the data set as an output data set.

KEYREF=LIST
Refers to a keyword in the build map or architecture definition. Use LIST to identify the listing data set.

DDNAME=DGIPRINT
Specifies the ddname. For the listing you have to use DGIPRINT.

DFLTTYP=*type*
Indicates the SCLM type for translator outputs. Use the name you have selected for the listing of this translator.

RECFM=FBA
Specifies the record format. Use record format FBA for DGIPRINT.

LRECL=121
Specifies the logical record length. Use 121 for DGIPRINT.

RECNUM=*nnn*
Specifies the expected number of records for the listing. (It is suggested to specify 500.)

PRINT=Y
Indicates that the data set is to be printed.

FLMALLOC macro

This macro defines the generated data structures or control block source.
Specify one macro for each output data set to be created by the translator.

IOTYPE=O,
Identifies the data set as an output data set.

KEYREF=OUT*x*
Refers to a keyword in the build map or architecture definition. Use OUT*x* for the maps and data structures.

DDNAME=*name*
Specifies the ddname. You may use any name. You have to specify this name in the request file too. This name can be passed to the request file via an additional option. See the OPTIONS parameter of the FLMTRNSL macro.

DFLTTYP=*type*
Indicates the SCLM type for translator outputs. Use the name you specified in the FLMTYPE macro for this object type.

RECFM=FB
Specifies the record format. Use record format FB.

LRECL=*nnn*
Specifies the logical record length. For GDDM-IMD or CSP/AD export data sets use 256. In all other cases (including external source format for CSP/AD), use 80.

RECNUM=*nnn*
Specifies the expected number of records of the generated output.

The SCLM project definition needs to be assembled. Refer to the *SCLM Guide and Reference* for details.

Step 4: Preparing build request files

To prepare build request files you can use the examples from the library SDF2.V1R4M0.SDGISAM for information about request files.

The names of the example request files are:

DGIIPNL*n* For object type panel, for *n*= 2 or 3
DGIIGRP*n* For object type panel group, for *n*= 1, 4, or 5

Where *n* is 1 for CICS/BMS, 2 for IMS/MFS, 3 for ISPF, 4 for GDDM-IMD, or 5 for CSP/AD.

The request files are built like for the batch invocation of SDF II. Refer to “Preparing a request file” on page 49 for a detailed explanation of request files.

If you want to set any variables in a request file to blank, you need to set these variables explicitly. For uninitialized variables a value of X'00' will be assumed.

Step 5: Adapting user exit routine DGILXLI

To define your libraries in this user exit routine, follow these steps:

1. Search for these comment lines:

```
/*********************************************************************
* Definition of supported libraries
*
```

2. Add lines for each SCLM controlled library, such as:

```
libs=n                          /* The library counter            */
name.x="libraryname"            /* The library name               */
type.x="SCLM"                   /* The library type is SCLM       */
proj.x="projname"               /* SCLM project name              */
grp.x="groupname"               /* SCLM group name                */
```

Where:

n is the total number of library specifications that follow.

x is 1 for the first library, 2 for the second, and so forth.

libraryname is the library name that is to be used on the Specify Libraries panel; it can be chosen freely. It is, however, suggested to choose identical names for library name (`name.x`) and project name (`proj.x`) without quotes.

The library name must be exactly as entered on the Specify Libraries panel, including any quotes if present.

projname is the SCLM project name.

groupname is the SCLM group name of the user level in the library hierarchy. This group name identifies the path up the hierarchy that SDF II follows to search for the object to be edited. SDF II takes the first occurrence of the object.

SDF II objects will be stored in this group. Usually, this is a group assigned to an individual programmer. If so, you may use the REXX built-in function `USERID` in the form `"USERID"( )`.

Additionally, you may store the following variables in the shared pool:

`ILVUEASK` The access key
`ILVUEATC` The authorization code
`ILVUECGC` The change code

SDF II passes the values defined for these variables to SCLM services.

Step 6 (optional): Adapting user exit routine DGILXOL

To improve the performance in the List Objects dialogs, you may:

- Delete the ALLOCs and FREEs from this user exit routine and
- In the invocation EXEC at the end of the assignment section, set the variable `sclm_alloc` to `Y`.

Step 7 (optional): Adapting user exit routines DGILXOR and DGILXOD

You can adapt these routines to allow renaming or deleting objects in SCLM controlled libraries.

Note: Renaming and deleting is not supported in the shipped version of SDF II.

Externally controlled libraries

SDF II provides user exits for controlled libraries. User exit routines for these exits must be written by the user. Examples of user exit routines from which the user can start are explained in “User exit routines for libraries” on page 29. You will find an example for each of these user exit routines in SDF2.V1R4M0.SDGISAM.

After the system programmer has set up the object libraries and, for controlled libraries, adapted the user exit routines, each user who needs to access libraries has to specify them on the Specify Libraries panel of SDF II. To get there, select option **8** on the Select an SDF II Function panel.

For more information about specifying libraries refer to *SDF II General Introduction*, chapter *Setting up your SDF II session*, under *Specifying libraries*.

Objects that are stored in externally controlled libraries are edited in SDF II. To allow editing of SDF II objects from your library system, SDF II can be called via the invocation interface. For more information see “Using the invocation interface” on page 39.

User exit routines for libraries

Examples of user exit routines are shipped with SDF II. REXX EXEC examples of the following routines are in the sample library SDF2.V1R4M0.SDGISAM:

DGILXLI	Check, initialize, and release a library.
DGILXOL	List the objects in the library.
DGILXOD	Delete an object from a library.
DGILXOR	Rename an object in a library.
DGILXIO	Read an object from or write it to a library.

DGILXLI

This user exit routine is called:

- To check a library name and determine the library type when it is entered on the Specify Libraries panel
- To initialize a library when it is addressed by SDF II
- To release a controlled library when the Specify Libraries dialog is entered or SDF II is left

The shipped version of the DGILXLI EXEC always returns with a return code of 0 and SDF II operates as in previous releases.

To use controlled libraries, you have to define your controlled libraries in the DGILXLI user exit routine.

You can customize DGILXLI for ISPF libraries as described in “Step 3 (optional): Adapting user exit routine DGILXLI” on page 22. For controlled libraries, you need to customize it as described in “Step 5: Adapting user exit routine DGILXLI” on page 27. If you use externally controlled libraries, choose an appropriate value for *type.x*. In the examples provided by SDF II, EXT is used for externally controlled libraries. If you choose different values, you have to adapt the other user exit routines accordingly.

Figure 5 lists the variables for the user exit routine DGILXLI:

Figure 5 (Page 1 of 2). Variables for user exit routine DGILXLI

Name	Type	Length	Description
Arg(1)	Arg	1	Request. One of: C Check the library specification. I Initialize/set up the library for use. F Free/release the library. **Note:** Before calling this user exit with request F, SDF II will perform LMCLOSE and LMFREE for all associated active data sets.
ILVUELIB	in	any	The library specification as entered on the Specify Libraries panel.

Figure 5 (Page 2 of 2). Variables for user exit routine DGILXLI

Name	Type	Length	Description
ILVUELUD	i/o	any	Any data related to the library used for communication between library user exit routines. It is not used by SDF II. The data is cleared by SDF II after a DGILXLI F or C request.
ILVUEPWD	in	8	The password as specified on the Specify Libraries panel.
ILVUEPNM	out	8	SCLM project name. Must be set for an SCLM controlled library for an **I** request.
ILVUEGN1	out	8	Group name of lowest SCLM group. Must be set for an SCLM controlled library for an **I** request.
ILVUEMSG	out	8	Message ID. In this variable the user exit may return a message ID. The specified message will be displayed by SDF II.
ILVUEASK	out	16	The access key. It is used during one SDF II session. SDF II passes the value defined for this variable to SCLM services.
ILVUEATC	out	8	The authorization code. It is used during one SDF II session. SDF II passes the value defined for this variable to SCLM services.
ILVUECGC	out	20	The change code. It is used during one SDF II session. SDF II passes the value defined for this variable to SCLM services.
Rc	out		Return Codes 0 It is a PDF library. 1 It is an SCLM controlled library. 2 It is an externally controlled library. 8 Invalid library specification. 32 Fatal error.

Note:

in	Input for the user exit routine. Stored in the shared pool.
out	Output from the user exit routine. Stored in the shared pool.
i/o	Input and output. Stored in the shared pool.
Arg	Input via parameter.

DGILXOL

This user exit routine is called:

- Once for each library, to initialize the required data sets when the List Objects dialog is entered or when the Refresh command is entered.
- Once for each object type within a library to get a list of objects. The list of objects has to be returned in a file with the ddname DGILXOL or an ISPF table named DGILXOL.
- Once for each library, to free the data sets when leaving the List Objects dialog.

Figure 6 lists the variables for the user exit routine DGILXOL:

Figure 6. Variables for user exit routine DGILXOL

Name	Type	Length	Description
Arg(1)	Arg	1	Request. One of: I Initialize/allocate required data sets. L Create list. F Free data sets.
ILVUENAM	in	any	Requested object names. Specifies the object names or generic names. • An * represents zero or more arbitrary characters at that position. • A % represents exactly one arbitrary character at that position.
ILVUETYP	in	any	Requested object type. DGIPNL Panel DGIGRP Panel group DGIPST Partition set DGITBL AID table DGIOCT Control table
ILVUELIB	in	any	The library specification as entered on the Specify Libraries panel.
ILVUELUD	i/o	any	Any data related to the library used for communication between library user exit routines. Not used by SDF II.
ILVUELID	i/o	8	The library ID returned from LMINIT (not valid for SCLM controlled libraries). If blank, no LMINIT has been performed.
ILVUEORG	i/o	2	Organization as returned by the PDF service LMOPEN. If blank, the data set is not open.
ILVUEOPM	i/o	1	The library status: blank The library is not open. I The library is open for input. O The library is open for output.
ILVUEPWD	in	8	The password as specified on the Specify libraries panel.
ILVUEMSG	out	8	Message ID. In this variable the user exit may return a message ID. The specified message will be displayed by SDF II.
Rc	out		Return Codes 0 ISPF table DGILXOL created. 1 File with ddname DGILXOL created. 8 No object found. 16 Library not assigned. 32 Unrecoverable error.

Note:

in Input for the user exit routine. Stored in the shared pool.
out Output from the user exit routine. Stored in the shared pool.
i/o Input and output. Stored in the shared pool.
Arg Input via parameter.

Figure 7 lists the contents of the ISPF table that is created on return code 0. It contains one row for each object as described.

Figure 7. Table DGILXOL - Library contents

Column	Length	Description
ILVULNAM	1-8	Name of the object
ILVULDAT	80	Contents of the first record of the SDF II object

Figure 8 lists the contents of the file that is created on return code 1. It contains one row for each object as described.

Figure 8. Layout of file DGILXOL - Library contents

Column	Description
1-8	Name of the object
10-17	Group name. Required only for SCLM controlled libraries.
19-26	DGIPNL Panel DGIGRP Panel group DGIPST Partition set DGITBL AID table DGIOCT Control table

DGILXOD

This user exit routine is called to delete an object from a controlled library when requested by a **d** (delete) command in the List Objects dialog.

Figure 9 lists the variables for user exit routine DGILXOD:

Figure 9 (Page 1 of 2). Variables for user exit routine DGILXOD

Name	Type	Length	Description
ILVUENAM	in	8	Object name
ILVUETYP	in	any	Object type DGIPNL Panel DGIGRP Panel group DGIPST Partition set DGITBL AID table DGIOCT Control table
ILVUELIB	in	any	The library specification as entered on the Specify Libraries panel.
ILVUELUD	i/o	any	Any data related to the library used for communication between library user exit routines. Not used by SDF II.
ILVUELID	i/o	8	The library ID returned from LMINIT (not valid for SCLM controlled libraries). If blank, no LMINIT has been performed.
ILVUEORG	i/o	2	Organization as returned by the PDF service LMOPEN. If blank, the data set is not open.
ILVUEOPM	i/o	1	The library status: blank The library is not open. I The library is open for input. O The library is open for output.

Figure 9 (Page 2 of 2). Variables for user exit routine DGILXOD

Name	Type	Length	Description
ILVUEPWD	in	8	The password as specified on the Specify Libraries panel.
ILVUEMSG	out	8	Message ID. In this variable the user exit routine may return a message ID, which is displayed by SDF II.
Rc	out		Return Codes 0 Object deleted. 8 Object not found. 16 Library not assigned. 24 Delete not possible. 32 Fatal error.

Note:

in Input for the user exit routine. Stored in the shared pool.
out Output from the user exit routine. Stored in the shared pool.
i/o Input and output. Stored in the shared pool.
Arg Input via parameter.

DGILXOR

This user exit routine is called to rename an object in a controlled library when requested by a **r** (rename) command in the List Objects dialog.

Figure 10 lists the variables for user exit routine DGILXOR:

Figure 10 (Page 1 of 2). Variables for user exit routine DGILXOR

Name	Type	Length	Description
ILVUENAM	in	8	Object name
ILVUETYP	in	any	Object type DGIPNL Panel DGIGRP Panel group DGIPST Partition set DGITBL AID table DGIOCT Control table
ILVUELIB	in	any	The library specification as entered on the Specify Libraries panel.
ILVUELUD	i/o	any	Any data related to the library used for communication between library user exit routines. Not used by SDF II.
ILVUELID	i/o	8	The library ID returned from LMINIT (not valid for SCLM controlled libraries). If blank, no LMINIT has been performed.
ILVUEORG	i/o	2	Organization as returned by the PDF service LMOPEN. If blank, the data set is not open.
ILVUEOPM	i/o	1	The library status: blank The library is not open. I The library is open for input. O The library is open for output.
ILVUEPWD	in	8	The password as specified on the Specify Libraries panel.

Figure 10 (Page 2 of 2). Variables for user exit routine DGILXOR

Name	Type	Length	Description
ILVUENNM	in	any	New object name
ILVUEMSG	out	8	Message ID. In this variable the user exit may return a message ID. The specified message will be displayed by SDF II.
Rc	out		Return Codes 0 Object renamed. 4 Library already contains the specified new name. 8 Object not found. 16 Library not assigned. 24 Rename not possible. 32 Unrecoverable error.

Note:

in	Input for the user exit routine. Stored in the shared pool.
out	Output from the user exit routine. Stored in the shared pool.
i/o	Input and output. Stored in the shared pool.
Arg	Input via parameter.

DGILXIO

This user exit routine is called only for externally controlled libraries:

- To lock an object
- To unlock an object
- To open an object and return an ID
- To close an object.

For OPEN requests, the user exit routine has to issue the commands LMINIT and LMOPEN. With LMINIT the variable ILVUELID has to be set. The LMOPEN command sets the variable ILVUEORG. For an OR (open before read) request for an object in a partitioned data set, the user exit routine has to issue also the LMMFIND command to locate the member with the name ILVUENAM.

For a CW (close after write) request for an object in a partitioned data set, the user exit routine has to issue the LMMREP command. SDF II issues an LMCLOSE command, if ILVUELID and ILVUEOPM are set to any value other than blank. If ILVUELID is set, SDF II also issues the LMFREE command. If you wish to do the LMCLOSE/LMFREE under your control, set the variables ILVUEOPM and ILVUELID to blank and store these values in the shared pool.

Figure 11 lists the variables for user exit routine DGILXIO:

Figure 11 (Page 1 of 2). Variables for user exit routine DGILXIO

Name	Type	Length	Description
Arg(1)	in	any	Request. One of: OR Open before read. CR Close after read. OW Open before write. CW Close after write. ODEL Open before delete. CDEL Close after delete. OREN Open before rename. CREN Close after rename. ODIR Open for directory read. CDIR Close after directory read. U Unlock.
Arg(2)	in	any	Disposition L Lock the object. U Do not lock the object.
ILVUENAM	in	1-8	Object name
ILVUETYP	in	any	DGIPNL Panel DGIGRP Panel group DGIPST Partition set DGITBL AID table DGIOCT Control table
ILVUELIB	in	any	The library specification as entered on the Specify Libraries panel.
ILVUELUD	i/o	any	Any data related to the library used for communication between library user exit routines. Not used by SDF II.
ILVUELID	i/o	8	The library ID returned from LMINIT (not valid for SCLM controlled libraries). If blank, no LMINIT has been performed.
ILVUEORG	i/o	2	Organization as returned by the PDF service LMOPEN. If blank, the data set is not open.
ILVUEOPM	i/o	1	The library status: blank The library is not open. I The library is open for input. O The library is open for output.
ILVUEPWD	in	8	The password as specified on the Specify Libraries panel.
ILVUEMSG	out	8	Message ID. In this variable the user exit may return a message ID. The specified message will be displayed by SDF II.

Figure 11 (Page 2 of 2). Variables for user exit routine DGILXIO

Name	Type	Length	Description
Rc	out		Return Codes 0 OK. 8 Object not found. 16 Library not assigned. 32 Fatal error.

Note:

in	Input for the user exit routine. Stored in the shared pool.
out	Output from the user exit routine. Stored in the shared pool.
i/o	Input and output. Stored in the shared pool.
Arg	Input via parameter.

Chapter 4. Interfacing with SDF II

This chapter contains General-Use Programming Interface and Associated Guidance information.

This chapter describes how to:

- Call SDF II from an ISPF menu
- Use the SDF II invocation interface
- Run SDF II in batch
- Write the related field user exit routines
- Write a generation user exit routine

Calling SDF II from an ISPF menu

Normally, SDF II is called by using the **sdf2** or the **sdf2c** command. You can add these commands to any ISPF menu, such as the PDF primary option menu, to make them a selectable item. To do so, follow this procedure for each command (sdf2 or sdf2c) that you want added to the ISPF menu:

1. Choose an option that is not yet used in the menu.
2. Insert a line in the menu that describes the option.
3. Insert a line in the)PROC section of the panel for each command:
   For the **sdf2** command `CMD(SDF2INV REQUEST(SDF2) [OPTION(`*option*`)]) NEWAPPL(DGI)`
   For the **sdf2c** command `CMD(SDF2INV REQUEST(SDF2C) [OPTION(`*option*`)]) NEWAPPL(DGI)`
4. Start SDF II with `NEWAPPL(DGI)`.

For *option* insert the value that is to be selected on the first panel.

Note: If you want to add the **sdf2c** command to any ISPF menu but protect it from general use, make sure this panel is accessible only to the system programmer.

Figure 12 on page 38 shows the SDF II primary option menu, DGI@PRIM, which already contains the three commands for calling SDF II. It is contained in the SDF II panel library.

```
)ATTR DEFAULT(¢;])
 ; TYPE(NT)  SKIP(ON)
 ¢ TYPE(PT)  SKIP(ON)
 $ TYPE(FP)  SKIP(ON)
 { TYPE(PS)  SKIP(ON)
   TYPE(ET)  SKIP(ON)
 } TYPE(SAC) SKIP(ON)
 ] TYPE(NEF) PAD(NULLS)
)BODY
;          ¢    SELECT AN SDF II MAIN FUNCTION       ;                        ;
$Command ===>]ZCMD                                                            ;
;
}0  {ISPF PARMS           ; Specify terminal and user parameters
;
}1  {SDF II FUNCTIONS     ; Run editors and utilities
;
}2  {SDF II CUSTOMIZATION ; Customize device table and perform translation
;
}X  {EXIT                 ; Terminate ISPF using list and log defaults
;
;
;
;
                Licensed Materials - Property of IBM (R)
   5665-366 (C) Copyright IBM Corp. 1987, 1995.  All rights reserved.
     IBM is a registered trademark of International Business Machines Corp.
)INIT
 .HELP=ISP00003
 &ZPRIM = YES          /* This is a primary option menu           */
 &ZHTOP = ISP00003     /* Tutorial table of contents for this appl*/
 &ZHINDEX = ISP91000   /* Tutorial index - 1st page for this appl */
)PROC
 &X = TRUNC (&ZCMD,'.')            /* required to get lower level   */
 &X = .TRAIL                       /* SELECT options passed to SDF II*/
 &Y = 'OPTION(&X)'                 /* SET ALL OPTION PARM.           */
 IF (&X = ' ')                     /* IF NO SUFFIX ENTERED           */
   &Y = ' '                        /* BLANK OUT OPTION PARM          */
 &ZSEL = TRANS( TRUNC (&ZCMD,'.')
              0,'PANEL(ISPOPTA) NEWAPPL(DGI)'
              1,'CMD(%SDF2INV REQUEST(SDF2)  &Y) NOCHECK NEWAPPL'
              2,'CMD(%SDF2INV REQUEST(SDF2C) &Y) NOCHECK NEWAPPL'
            ' ',' '
              X,'EXIT'
              *,'?')               /* INVALID OPTION                 */
)PNTS
 FIELD(ZPS00001) VAR(ZCMD) VAL(0)
 FIELD(ZPS00002) VAR(ZCMD) VAL(1)
 FIELD(ZPS00003) VAR(ZCMD) VAL(2)
 FIELD(ZPS00004) VAR(ZCMD) VAL(X)
)END
```

Figure 12. SDF II primary option menu DGI@PRIM

Using SDF II with different languages

If you have installed more than one language, you may use the parameter `LANGUAGE` of the invocation routines to select a particular language. Call the routines as follows:

```
SDF2    parm [LANGUAGE(language)]
SDF2C   parm [LANGUAGE(language)]
SDF2INV parm [LANGUAGE(language)]
```

Where *parm* is any other parameter you require and *language* can be one of the following (valid abbreviations are shown in bold):

no argument	The default as defined in SDF2INV
English	English NLS feature
Japanese	Japanese NLS feature
German or **D**eutsch	German NLS feature
Spanish or **Es**panol	Spanish NLS feature
Swiss or **Sc**hweiz	Swiss German NLS feature.

If you call SDF2INV from within an ISPF environment, for example, from an ISPF menu, *language* must be one of the language identifiers EU, JN, DU, DS, or EP.

The default language can be changed in the SDF2INV routine. You have to set the variable `sdf2_lang` to the language identifier of the default language.

Using the invocation interface

SDF II provides an invocation interface, which can be used to invoke a selected function from any ISPF dialog. When you invoke a function, you describe the request to be performed by SDF II. On return, SDF II provides completion information. Information is transmitted to and from SDF II by means of the invocation interface table, an ISPF table with the default name DGIIFTAB.

The invocation interface table

The invocation interface table is a temporary table. It must be open when SDF II is called. Each row in the table identifies one request to SDF II. SDF II can handle multiple rows in one table and will process the requests one after the other, without returning control to the calling program. Only when the end of the ISPF table is reached, is control returned to the calling program. Depending on the request, SDF II may add rows to the table.

Two types of errors may occur in the table: row errors and invalid parameters. Rows that cannot be interpreted are ignored. When passing invalid parameters, the appropriate dialog panel is displayed with the appropriate error message.

Values that could not be set by using the ISPF table are taken from the profile.

Figure 13 lists the columns of the invocation interface table and their contents. Unless otherwise indicated, a column applies to all types of requests.

Figure 13 (Page 1 of 3). The columns of the invocation interface table (DGIIFTAB)

Column	Length	Description
IIVROW	1	The type of the row: V To identify a request row. When SDF II reads sequentially through the ISPF table, it performs the requests of only these rows. Rows with request types other than V are ignored. Z To identify a row that has been added by SDF II after successful completion of a request. These rows can be used by the invoker for further processing. They identify the action taken by SDF II. In response to a request, SDF II may add any number of rows. These rows are discussed in more detail later.
IIVREQ	1	The type of the request that should be processed: E Edit an object or prototype. C Copy an object. D Delete an object. G Generate an object. M Import an object. O Print the online reference. P Print an object. T Test an object or prototype. A Construct a panel. X Extract a panel. F Modify a panel.
IIVNAM	≤8 ≤46	The name of the object to be processed. The name of the object to be imported.
IIVTYP	1	The type of object to be processed: P Panel G Panel group S Partition set A AID table B Control table W Prototype The type of object to be imported: 1 CICS/BMS macros 2 IMS/MFS utility control statements 3 ISPF panel 4 GDDM-IMD export data set 5 CSP/AD export data set 6 SDF/CICS dump data set 7 External source format This column is not used for requests to construct a panel or to print the online reference.
IIVLIB	≤46	This identifies the library in which the object is stored or is to be stored. If the library identifier is a single character, SDF II assumes that it is the library identifier assigned in the Specify Libraries dialog. Otherwise, SDF II assumes that it is a library name of the form PROJECT.GROUP. In this case, SDF II looks for the 1-digit library identifier defined in the Specify Libraries dialog. For some requests, such as to generate or print an object, the library identifier can be an asterisk (*). This column is not used for requests to edit a prototype, test a prototype, or print the online reference.

Figure 13 (Page 2 of 3). The columns of the invocation interface table (DGIIFTAB)

Column	Length	Description
IIVONAM	8	This field has a different meaning for each of these requests: **EDIT** A second object name. When you create a new object, this is the name of the skeleton object. If you do not want to use a skeleton object, specify the value X'00'. **COPY** The name of the new object. **MODIFY** The name of the new object. **CONSTRUCT** The name of an ISPF table that contains the panel elements to be used. The default name is DGIU5TAB. See Figure 15 on page 46 for the structure of this ISPF table.
IIVOLIB		This field has a different meaning for each of these requests:
	≤46	**EDIT** The identifier of the second library. If the library identifier is a single character, SDF II assumes that it is the library identification assigned in the Specify Libraries dialog. Otherwise, SDF II assumes that it is a library name of the form PROJECT.GROUP. In this case, SDF II looks for the 1-digit library identifier defined in the Specify Libraries dialog. When you create a new object, this is the library identifier of the skeleton object. If you do not want to use a skeleton object, specify the value X'00'.
	≤17	**COPY** The library identifier of the new object. **MODIFY** The library identifier of the new object.
	15	**CONSTRUCT** The 15 characters of the column are made up of: • 3 characters for the depth of the panel in lines. It defaults to the depth defined in the device table. • 3 characters for the width of the panel in characters. It defaults to the width defined in the device table. • one character, which specifies: 0 Include the command line in the panel. 1 Exclude the command line from the panel. This applies only to ISPF panels. • 8 characters for the help panel name. This applies only to ISPF and CSP/AD panels.
IIVDEV	8	This is the device type for these requests: **EDIT** To be used when you create a new object without using a default object. It is optional for ISPF. **CONSTRUCT** For any target system except ISPF, where it is optional. **EXTRACT** To select the formats to be processed. **MODIFY** To select the formats to be processed.

Figure 13 (Page 3 of 3). The columns of the invocation interface table (DGIIFTAB)

Column	Length	Description
IIVMOD	1 or 2	This field has a different meaning for each of these requests: **EDIT** The first character holds the option number of the first editor dialog to be invoked. (The second character is ignored.) **GENERATE** The first character holds the target system for which the object is to be generated: 0 Application prototype 1 CICS/BMS 2 IMS/MFS 3 ISPF 4 GDDM 5 CSP/AD The second character is ignored. **PRINT** The first character holds the destination of the SDF II output: 1 Standard printer 2 GDDM print utility 3 Input to DCF 4 DBCS printer 5 DBCS printer with field outlining The second character is ignored. **COPY** 00 copy object 10 copy and replace object 01 copy related objects 11 copy and replace related objects
IIVPRO	1	This column controls whether a dialog is to be presented to the user or to be processed immediately: B Do not display any panel of the dialog. G Do not display the panels listed below. P Display these panels with all their values inserted. G and P apply to: **EDIT** Identify Object panel or Identify Prototype panel **PRINT** Specify Print Utility Parameters panel **GENERATE** Identify Object for Generation panel and Specify Generation Parameters panel **CONSTRUCT** Specify Panel Construction Utility Parameters panel and Specify Panel Elements panel

Note: If you specify the value blank (' ') for any of the columns IIVNAM, IIVLIB, IIVONAM, IIVOLIB, or IIVDEV, the value from the profile is used instead. If you want to set any of these columns to blank, specify the value X'00'.

When you have prepared the ISPF table, call SDF II with:

```
ISPEXEC SELECT CMD (SDF2INV REQUEST(SDF2I) [OPTION(tablename)]) NEWAPPL(DGI)
```

Where:

tablename is the name of the invocation interface ISPF table. The default is DGIIFTAB.

Figure 14 on page 44 is an example of an EXEC. It creates the ISPF table and invokes SDF II. It can be called in an active ISPF environment.

```
/* ------------------------ REXX ---------------------------------- */
/* Create the ISPF table DGIIFTAB and invoke SDF II                 */
/* ------------------------------------------------------------------ */
   table = 'DGIIFTAB'              /* Function Table Name          */
   names = ,                       /* Column Names                 */
           'IIVROW',
           'IIVREQ',
           'IIVTYP',
           'IIVNAM',
           'IIVLIB',
           'IIVONAM',
           'IIVOLIB',
           'IIVDEV',
           'IIVMOD',
           'IIVPRO'
/* ------------------------------------------------------------------ */
/* Build ISPF Table                                                 */
/* ------------------------------------------------------------------ */
   address ISPEXEC                /* address ISPEXEC interface      */
   "CONTROL ERRORS RETURN"        /* return to REXX on errors       */
   "TBCREATE" table "NAMES("names") NOWRITE REPLACE" /* create table*/
   if rc>4 then signal error1     /* if create fails: error message */
/* ------------------------------------------------------------------ */
/* Fill Table Variables                                             */
/* ------------------------------------------------------------------ */
   iivrow  = "V"                  /* it is a Request row            */
   iivreq  = "E"                  /* Edit an Object                 */
   iivnam  = "MYPAN"              /* Object Name is MYPAN           */
   iivtyp  = "P"                  /* it is a Panel                  */
   iivlib  = "1"                  /* it is on Library 1             */
   iivonam = "MYDEFP"             /* use Default Panel MYDEFP       */
   iivolib = "1"                  /* Default Panel is on Library 1  */
   iivpro  = "G"                  /* do not prompt for Parameter    */
/* ------------------------------------------------------------------ */
/* Fill Table, call SDF II and discard Table                        */
/* ------------------------------------------------------------------ */
   "TBADD" table                  /* add the row                    */
   if rc>0 then signal error3     /* if tbadd  fails: error message */
   "SELECT CMD(SDF2INV REQUEST(SDF2I)) NEWAPPL(DGI)" /* call SDF II */
   irc=rc                         /* save retcode                   */
   "TBEND" table                  /* discard Table                  */
   exit irc                       /* exit program                   */
/* ------------------------------------------------------------------ */
/* Error Exits                                                      */
/* ------------------------------------------------------------------ */
   error1:
           say 'Return Code' rc 'from TBCREATE' table
           exit 12                /* exit program                   */
   error3:
           say 'Return Code' rc 'from TBADD' table
           exit 12                /* exit program                   */
/* ------------------------------------------------------------------ */
```

Figure 14. REXX example for the SDF2 invocation interface

When SDF II has successfully completed a request, it may add one or more rows to the ISPF table, or it may add none. These rows are identified by a row type of Z.

They immediately follow the request row, which is identified by a row type of V. The contents of these rows depend on the request, as follows:

Edit Each time you save the object you are editing, a row is added to the ISPF table. This row contains the fields:

IIVNAM	The name under which the object was saved
IIVTYP	The type of the object
IIVLIB	The library in which the object was saved.

Copy A row is added that contains the fields:

IIVNAM	The name of the target object
IIVTYP	The type of the object
IIVLIB	The library to which the object was copied.

Delete A row is added that contains the fields:

IIVNAM	The name of the deleted object
IIVTYP	The type of the object
IIVLIB	The library from which the object was deleted.

Generate For each output file created by the generation utility, a row is added to the ISPF table. This row contains the fields:

IIVNAM	The name of the generated object
IIVTYP	The type of the object
IIVLIB	The library in which the generated object was saved
IIVONAM	The member name of the file containing the generated output
IIVOLIB	The data set name (or *ddname*) of the file containing the generated output.

Construct
Each time you save the panel that you are constructing, a row is added to the ISPF table. This row contains the fields:

IIVNAM	The name of the constructed object
IIVTYP	The type of the object; it is always P (for panel)
IIVLIB	The library in which the constructed object was saved.

Import Each time an object is saved during import, a row is added to the ISPF table. This row contains the fields:

IIVNAM	The name of the imported object
IIVTYP	The type of the object
IIVLIB	The library in which the imported object was saved.

Modify A row is added that contains the fields:

IIVNAM	The name of the target object
IIVTYP	The type of the object
IIVLIB	The library to which the object was copied.

No entries are added to the table in response to requests to print the online reference, to print an object, or to test an object or prototype.

The construction utility table

For panel construction, additional information is transmitted to SDF II by means of this second ISPF table, whose default name is DGIU5TAB. You supply this name, or any name you choose, in the column IIVONAM of the invocation interface table (default name is DGIIFTAB).

The columns of the construction utility table contain basically the same information as the lines of the Specify Panel Elements panel. Column IUV5TYP, however, is not displayed on this panel. It is used only by the invocation interface.

For more information refer to the SDF II online reference.

Figure 15 lists the columns of the construction utility table and their contents.

Figure 15 (Page 1 of 2). The columns of the construction utility table (DGIU5TAB)

Column	Length	Description
IUV5TYP	1	This is the type of the panel element. The following values are possible: I — This identifies an ISPF procedural section line. Only column IUV5DES is used, which then contains the procedural section line. Table columns of this type are added to the ISPF procedural section of the panel. This is valid for the ISPF target system only. *other* — Any other character identifies a panel element. All of the following columns are used.
IUV5RFD	≤65	This is the name of the related field, which is passed to the user exit routines (see section "Using predefined panel elements" on page 69). You can enter qualified names, such as COLLECT.DATE. Each part of the name can be up to 32 characters long.
IUV5NAM	31	This is the field name used in SDF II. If left blank, it defaults to the last level of the qualified name in the IUV5RFD column. The default name must be unique within the panel and must conform to the syntax of the programming languages that SDF II supports. Otherwise, SDF II leaves this column blank.
IUV5FFO	8	This is the field format. The following values are possible: EBCDIC — The field contains only EBCDIC characters. DBCS — The field contains double byte character set (DBCS) characters. MIXED — The field contains EBCDIC and DBCS characters enclosed by SO/SI characters. MSUPRESS — The same as MIXED, but SO/SI characters are suppressed (no character position is required for them on printers). DEFMIXED — The device default is used. In the case of MFS only, this allows an application program to specify the field format attribute dynamically. The default value depends on the available markers and on the device type.
IUV5LEN	4	This is the length of the data field. It is optional.
IUV5OCC	3	This is either the occurrence number of a repeat format or the dimension of an array. It is optional.

Figure 15 (Page 2 of 2). The columns of the construction utility table (DGIU5TAB)

Column	Length	Description
IUV5VER	1	This is the direction of the array or repeat format. If no occurrence number is specified, it must be blank. The following values are possible: 1 For a vertical array or repeat format. *blank* For a horizontal array or repeat format. This is the default.
IUV5PRO	≤20	This is the prompt text associated with the field.
IUV5DES		This column contains:
	80	For IUV5TYP=I, the contents of an ISPF procedural section line.
	64	Otherwise, the description of the field.

Data extraction and modification utilities

The sample DGIUX60C in the sample library shows how to specify a user exit for the data extraction and modification utilities.

Variables provided in the user exits

Figure 16 lists the variables that are set in the ISPF SHARED POOL before the user exit is called.

Figure 16 (Page 1 of 2). General output variables

Name	Length	Description
The following shared pool variables are set before processing of an object starts:		
IUV62FUN	1	Function 1 Extract data. 2 Copy object and modify contents.
IUV62NAM	8	The name of the object.
IUV62LIB	1	The library ID from where the object has been read.
IUV62TYP	1	The type of the object: P Panel
IUV62NNA	8	The name of the new object.
IUV62NLI	8	The library ID where the new object will be stored.
The following variable is set before every user exit call:		
IUV62REQ	3	The type of the item to process PPB Begin of panel processing PPE End of panel processing PFB Begin of format processing PFE End of format processing PEV Variable field PEC Constant field PEB Background PEP Pull down choice

Figure 16 (Page 2 of 2). General output variables

Name	Length	Description
The following variables are set before the user exit is called with IUV62REQ='PFB':		
IPVFODVT	8	Device type of the format instance.
IPVFOFON	8	Format name (applicable for MFS panels only).
IPVFOPNB	*num*	Number of physical pages (applicable for MFS panels only, is 1 for all other target systems).
IPVFOPNN	*num*	Index number of physical page (1 ... IPVFOPNB) (applicable for MFS panels only, is 1 for all other target systems).
IPVFODEP	*num*	Format depth. Is 0 if the device name (IPVFODVT) identifies a stream device.
IPVFOWID	*num*	Format width.
The following variables are set when the user exit is called with IUV62REQ='PEx':		
IPVEONAM	32	The name of the field. Is blank for background items.
IPVEOLIN	*num*	Vertical position of the item. Is 0 if the device name (IPVFODVT) identifies a stream device.
IPVEOCOL	*num*	Horizontal position of the item.
IPVEOINC	8	If the item is part of an include panel this field contains the name of the include panel. **Note:** No changes can be performed within the include panel. The include panel has to be processed as separate object.
IPVEOLIX	8	If the item is part of a repeat format this variable contains the index. Otherwise it is 0.
IPVEOARX	8	If the field is part of an array this variable contains the index. Otherwise it is 0.
IPVEOFFO	1	Field format of the item: E EBCDIC M Mixed D DBCS (possible for fields only) S Mixed suppress (MFS panels only) Background can only be EBCDIC or MIXED.
IPVEOLEN	*num*	The length of the item: • For fields this is the length of the field. • For background this is the length from the 1st position (IPVEOCOL) up to the next field or to the end of the line, whichever is shorter.
IPVEOTXT	any	The text associated with the item. It contains: • For variable fields, the initial value, or the literal in case of MFS panels • For constant fields and for background, the associated text. The following rules apply to the text: • For background this is the text from the 1st position (IPVEOCOL) up to the next field or to the end of the line, whichever is shorter.

Additional variables provided in the user exit for the copy utility: Figure 17 on page 49 lists variables that can be modified by the user exit within the SHARED POOL.

Figure 17. Input variables

Name	Length	Description
IPVENFFO	1	Field format of the item. Can be set for Fields (Variable & Constant) and background when the device the instance is defined for supports it. E EBCDIC M Mixed D DBCS (possible for fields only) S Mixed suppress (MFS panels only) (Background can only be EBCDIC or MIXED)
IPVENTXT	any	The text associated with the item. Contains: • For variable fields, the initial value text. • For constant fields and for background, the text.

Running SDF II in batch

Some functions may need to be run in a batch environment. You can perform these SDF II tasks in a batch environment:

- Generate objects
- Import objects
- Print objects
- Print the online reference
- Construct panels
- Translate
- Copy related objects

Preparing a request file

To process a request for batch processing, you first need to prepare a request file. In the request file, include each request and its parameters. You can put more than one request in a request file. For each request, identify the object involved and any additional parameters required for the request to be successfully processed. If an error occurs, an entry is written to the ISPF log data set, and the request is terminated.

The request file must have a record length of 80 bytes and be in fixed record format. Each line of the request file has the form:

varname = '*value*'

Where:

varname is the name of a variable.

value is the value to be assigned to *varname*. The value must be in single quotation marks and must not contain a semicolon.

Note: For check boxes, specify **Y** to select, **N** to deselect.

When you include more than one request in a request file, indicate the start of the next request by entering an asterisk (*) in column 1, and leave the rest of the line blank. You can find examples of request files in “Batch request examples” on page 51.

Running the DGIIJBAT job

You can use the example job DGIIJBAT or the DGIIJBAT EXEC to process a request in the background. The example job is in the job control library SDF2.V1R4M0.SDGIJCL, the EXEC is in SDF2.V1R4M0.SDGISAM.

You will have to customize the job control statements in the example job or EXEC for your environment.

Make sure that batch processing and online processing do not use the same ISPF profile at the same time, and that the ISPLOG data set is not shared. A possible way is to provide a TSO prefix different to the user ID and to preallocate the ISPLOG data set. The SDF2INV routine will then allocate the profile to *prefix*.ISPF.ISPPR*xx* if you have not changed it. (Where *xx* is the language identifier.)

By using this general procedure you can invoke any of the functions that are available in batch.

Batch request examples

With the assignment statements in the request file, you assign values to ISPF variables. You can assign any value to the ISPF variables that you can enter from the relevant panels in the dialog.

Make sure that the ISPF profile you created with the profile editor or in the Specify Libraries dialog is available to the batch job.

If you are using a new profile or a profile where new library identifiers are required, you can set a library identifier `ILV10L0`*x* (*x*=1 through 9) in the request file. For example:

```
ILV10L02="'PROJECT.GROUP'"
```

will let you use the new library PROJECT.GROUP by setting the variable `IIVLIB` to 2. (Be careful to enter both single and both double quotes.)

Note: The figures in the examples of this section show the variable names for the panel input fields and not the input fields themselves.

Example 1: Generate a PL/I data structure and CICS/BMS macros

Figure 18 is a request file for generating a PL/I data structure and CICS/BMS macros for the CICS/BMS panel MYPAN. The generated output is to be stored in the partitioned data sets APPL1.DSECT.LIB and APPL1.MACRO.LIB.

```
IIVREQ = 'G'             /* requests generation                          */
IIVNAM = 'MYPAN'         /* identifies the object name                   */
IIVLIB = '*'             /* uses search order                            */
IIVTYP = 'P'             /* identifies object as panel                   */
IIVMOD = ''              /* generates for object's target system         */
ICV10GDS = 'Y'           /* generates a data structure                   */
ICV10GCB = 'Y'           /* generates a control block source             */
ICV10DLA = 'PLI'         /* identifies programming language              */
ICV10DAL = 'N'           /* generates unaligned data structure           */
ICV10DGR = 'N'           /* generates no GRAPHIC declarations            */
ICV10DTR = 'N'           /* generates no TRIGRAPH declarations           */
ICV10DUE = ''            /* invokes no user exit routine                 */
ICV10DOL = "'APPL1.DSECT.LIB'" /* identifies pds for generated data str. */
ICV10DOD = ''            /* identifies no ddname                         */
ICV10CDT = '*'           /* generates control block source for all devices */
ICV10CUE = ''            /* invokes no user exit routine                 */
ICV10COL = "'APPL1.MACRO.LIB'" /* identifies for control block source    */
ICV10COD = ''            /* identifies no ddname                         */
```

Figure 18. Request file to generate a PL/I data structure and CICS/BMS macros

Figure 19 and Figure 20 on page 53 show the panels and the names of the dialog variables that you have set with the request file in this example.

The names of the variables shown in Figure 19 for example 1 are the same for all requests for generation. This figure applies to examples 1 through 6 and will, therefore, not be repeated for examples 2 through 6.

```
DGICE00              IDENTIFY OBJECT FOR GENERATION
Command ===>

Identify the source object
  Name  . . . . . . . . . IIVNAM
  Library . . . . . . . . IIVLIB
  Type  . . . . . .  IIVTYP G. PANEL GROUP           S. PARTITION SET
                           P. PANEL

Specify the target system here only when you want to generate
for an other target system than specified for the object.
  Target system . .  IIVMOD blank Take target system from object
                              0. PROTOTYPING
                              1. CICS/BMS
                              2. MFS
                              3. ISPF
                              4. GDDM
                              5. CSP
```

Figure 19. Variables for identifying an object for generation

```
DGICE10              SPECIFY GENERATION PARAMETERS    MYPAN     Top of data
Command ===>
                                                                More:   +
Object name . . . . . : MYPAN
Object type . . . . . : PANEL
Target system . . . . : CICS/BMS

Enter '/' to select the generation function.
ICV10GDS   Generate data structure
ICV10GCB   Generate CICS/BMS macros

Specify the options and output libraries.
Then press ENTER to generate, or press END to exit.

Options for data structure
  Language  . . . . . . . ICV10DLA  ASM, C, COBOL, PLI, RPG
  Enter '/' to select option
ICV10DAL Alignment             Ignored for C language
ICV10DGR Graphic               DBCS panels only
ICV10DTR Trigraph              C language only
ICV10ADI SDF/CICS Compatible   COBOL language only

 Output library for data structure
   Dataset name or . . . . ICV10DOL
   DDname  . . . . . . . . ICV10DOD
   User exit . . . . . . . ICV10DUE

 Options for BMS macros
   Device type . . . . . . ICV10CDT   * for all devices

 Output library for BMS macros
   DATASET NAME or . . . . ICV10COL
   DDNAME  . . . . . . . . ICV10COD
   User exit . . . . . . . ICV10CUE
```

Figure 20. Variables for generation parameters for CICS/BMS

Example 2: Generate a COBOL data structure and IMS/MFS utility control statements

Figure 21 is a request file for generating a COBOL data structure and IMS/MFS utility control statements for the IMS/MFS panel MYPAN. The generated output is to be stored in the partitioned data sets APPL1.DSECT.LIB and APPL1.MACRO.LIB.

```
IIVREQ = 'G'             /* specifies generation                          */
IIVNAM = 'MYPAN'         /* identifies the object name                    */
IIVLIB = '*'             /* uses search order                             */
IIVTYP = 'P'             /* identifies object as panel                    */
IIVMOD = ''              /* use object's target system                    */
ICV12GDS = 'Y'           /* generates  a data structure                   */
ICV12GCB = 'Y'           /* generates a control block source              */
ICV12DLA = 'COBOL'       /* identifies programming language               */
ICV12DGR = 'N'           /* generates no GRAPHIC declarations             */
ICV12DUE = ''            /* invokes no user exit routine                  */
ICV12DOL = "'APPL1.DSECT.LIB'" /* identifies pds for generated data str.  */
ICV12DOD = ''            /* identifies no ddname                          */
ICV12IPS = 'Y'           /* includes partition sets                       */
ICV12IOT = 'Y'           /* includes control tables                       */
ICV12CUE = ''            /* invokes no user exit routine                  */
ICV12COL = "'APPL1.MACRO.LIB'" /* identifies pds for control block source */
ICV12COD = ''            /* identifies no ddname                          */
```

Figure 21. Request file to generate a COBOL data structure and MFS utility statements

Figure 22 on page 55 shows the names of the ISPF variables on the Specify Generation Parameters panel that you have set with the request file in this example.

```
DGICE12              SPECIFY GENERATION PARAMETERS                MYPAN
Command ===>
                                                                 More:    +
Object name . . . . . : MYPAN
Object type . . . . . : PANEL
Target system . . . . : MFS

Enter '/' to select the generation function.
ICV12GDS   Generate data structure
ICV12GCB   Generate MFS format set

Specify the options and output libraries.
Then press ENTER to generate, or press END to exit.

Options for data structure
  Language  . . . . . . . ICV12DLA    ASM, COBOL, PLI
  Enter '/' to select option
ICV12DGR  Graphic               DBCS panels only

Output library for data structure
  Dataset name or . . . . ICV12DOL
  DDname  . . . . . . . . ICV12DOD
  User exit . . . . . . . ICV12DUE

Options for MFS format set
ICV12IPS  Include partition sets
ICV12IOT  Include CTL tables

Output library for MFS format set
  DATASET NAME or . . . . ICV12COL
  DDNAME  . . . . . . . . ICV12COD
  User exit . . . . . . . ICV12CUE
```

Figure 22. Variables for generation parameters for IMS/MFS

Example 3: Generate an ISPF panel

Figure 23 is a request file for generating an ISPF panel for the ISPF panel MYPAN. The generated output is to be stored in the partitioned data set APPL1.PANEL.LIB.

```
IIVREQ = 'G'             /* requests generation                       */
IIVNAM = 'MYPAN'         /* identifies the object name                */
IIVLIB = '*'             /* uses search order                         */
IIVTYP = 'P'             /* identifies object as a panel              */
IIVMOD = ' '             /* generates for object's target system      */
ICV13GDS ='Y'            /* generates data structure                  */
ICV13GCP ='P'            /* generates panel                           */
ICV13DLA ='C'            /* identifies programming language           */
ICV13DGR = 'N'           /* generates no GRAPHIC declarations         */
ICV13DTR = 'N'           /* generates no TRIGRAPH declarations        */
ICV13DUE = '        '    /* invokes no user exit                      */
ICV13DOL = "'APPL1.DSECT.LIB'" /* identifies pds for data structure   */
ICV13DOD = '        '    /* identifies no ddname                      */
ICV13CUE = '        '    /* invokes no user exit routine              */
ICV13COL = "'APPL1.PANEL.LIB'" /* identifies pds for generated panel  */
ICV13COD = '        '    /* identifies no ddname                      */
```

Figure 23. Request file to generate an ISPF panel

Figure 24 shows the panel and the names of the ISPF variables that you have set with the request file in this example.

```
DGICE13              SPECIFY GENERATION PARAMETERS                MYPAN
Command ===>
                                                                 More:    +
Object name . . . . . : MYPAN
Object type . . . . . : PANEL
Target system . . . . : ISPF

Enter '/' to select the generation function.
ICV13GDS   Generate data structure
ICV13GCP   Generate ISPF panel

Specify the options and output libraries.
Then press ENTER to generate, or press END to exit.

Options for data structure
  Language  . . . . . . . ICV13DLA  ASM, C, COBOL, PLI
  Enter '/' to select option
ICV13DGR    Graphic              DBCS panels only
ICV13DTR    Trigraph             C language only


 Output library for data structure
   Dataset name or . . . . ICV13DOL
   DDname  . . . . . . . . ICV13DOD
   User exit . . . . . . . ICV13DUE

 Output library for ISPF panel
   DATASET NAME or . . . . ICV13COL
   DDNAME  . . . . . . . . ICV13COD
   User exit . . . . . . . ICV13CUE
```

Figure 24. Variables for generation parameters for ISPF

Example 4: Generate a GDDM-IMD export data set

Figure 25 is a request file for generating the GDDM-IMD export data set for the GDDM-IMD panel MYPAN. The data structure is to be coded in PL/I. The generated output is to be stored in the partitioned data set APPL1.ADMIFMT.LIB.

```
IIVREQ = 'G'             /* requests generation                          */
IIVNAM = 'MYPAN'         /* identifies the object name                   */
IIVLIB = '*'             /* uses search order                            */
IIVTYP = 'P'             /* identifies object as a panel                 */
IIVMOD = ' '             /* generates for object's target system         */
ICV14DLA = 'PLI'         /* identifies programming language              */
ICV14DGR = ' '           /* generates no GRAPHIC                         */
ICV14CUE = '        '    /* invokes no user exit routine                 */
ICV14COL = "'APPL1.ADMIFMT.LIB'" /* identifies pds for generated panel   */
ICV14COD = '        '    /* uses a ddname                                */
```

Figure 25. Request file to generate a GDDM-IMD export data set

Figure 26 shows the panel and the names of the ISPF variables that you have set with the request file in this example.

```
DGICE14              SPECIFY GENERATION PARAMETERS                MYPAN
Command ===>
                                                                More:     +
Object name . . . . . : MYPAN
Object type . . . . . : PANEL
Target system . . . . : GDDM

Options
  Language  . . . . . . . ICV14DLA   ASM, C, COBOL, PLI, PLDS
  Enter '/' to select option
ICV14DGR  Graphic              DBCS panels only

Output library
  Dataset name or . . . . ICV14COL
  DDname  . . . . . . . . ICV14COD
  User exit . . . . . . . ICV14CUE
```

Figure 26. Variables for generation parameters for GDDM

Example 5: Generate CSP/AD external source format

Figure 27 is a request file for generating external source format for the CSP/AD panel MYPAN. The generated output is to be stored in the partitioned data set APPL1.ESF.LIB.

To generate CSP/AD export data sets, specify 1 for ICV15CSE.

```
IIVREQ = 'G'             /* requests generation                     */
IIVNAM = 'MYPAN'         /* identifies the object name              */
IIVLIB = '*'             /* uses search order                       */
IIVTYP = 'P'             /* identifies the object as a panel        */
IIVMOD = ' '             /* generates for object's target system    */
ICV15CSE = '2'           /* generates external source format        */
ICV15CUE = '        '    /* invokes no user exit routine            */
ICV15COL = "'APPL1.ESF.LIB'" /* identifies pds for generated panel  */
ICV15COD = '        '    /* uses no ddname                          */
```

Figure 27. Request file to generate a CSP external source format

Figure 28 shows the panel and the names of the ISPF variables that you have set with the request file in this example.

```
DGICE15              SPECIFY GENERATION PARAMETERS              MYPAN
Command ===>

Object name . . . . . : MYPAN
Object type . . . . . : PANEL
Target system . . . . : CSP

Generation selection
  Generation of . . . . . ICV15CSE 1 - CSP/AD export data set
                                   2 - External source format

Options
  Device type . . . . . . ICV15CDT * for all devices
  Panel group name  . . . ICV15CPG

Output library
  DATASET NAME or . . . . ICV15COL
  DDNAME  . . . . . . . . ICV15COD
  User exit . . . . . . . ICV15CUE
```

Figure 28. Variables for generation parameters for CSP/AD

Example 6: Generate CICS/BMS partition set macros

Figure 29 is a request file for generating the BMS partition set MYPSET. The generated output is to be stored in the partitioned data set APPL1.MACRO.LIB.

```
IIVREQ = 'G'             /* requests generation                          */
IIVNAM = 'MYPSET'        /* identifies the object name                   */
IIVLIB = '*'             /* uses search order                            */
IIVTYP = 'S'             /* identifies object as a partition set         */
IIVMOD = ' '             /* generates for object's target system         */
ICV20DTY = '*'           /* generates control block source for all devices */
ICV20CUE = '        '    /* invokes no user exit routine                 */
ICV20COL = "'APPL1.MACRO.LIB'" /* identifies pds for generated part set  */
ICV20COD = '        '    /* uses no ddname                               */
```

Figure 29. Request file to generate a BMS partition set

Figure 30 shows the panel and the names of the ISPF variables that you have set with the request file in this example.

```
DGICE20               SPECIFY GENERATION PARAMETERS              MYPAN
Command ===>

Object name . . . . . : MYSET
Object type . . . . . : Partition Set
Target system . . . . : CICS/BMS

Options for BMS macros
  Device type . . . . . ICV20DTY * for all devices

Output library for BMS macros
  DATASET NAME or . . . . ICV20COL
  DDNAME  . . . . . ICV20COD
  User exit . . . . . ICV20CUE
```

Figure 30. Variables for generation parameters for CICS/BMS

Example 7: Import IMS/MFS utility control statements into SDF II

Figure 31 is a request file for importing IMS/MFS utility control statements. The input file has the name APPL1.MFSSRC.LIB(MYMFS). The imported objects are to be stored in your library 1. The copy library has the name APPL1.MFSCOP.LIB.

To import objects from any other target system, specify the appropriate value for IIVTYP.

```
IIVREQ = 'M'              /* requests import                              */
IIVTYP = '2'              /* identifies source as IMS/MFS utility stmts   */
IIVNAM = "'APPL1.MFSSRC.LIB(MYMFS)'" /* identifies source data set        */
IIVLIB = '1'              /* stores migrated object in library 1          */
IUV20CL = "'APPL1.MFSCOP.LIB'" /* identifies name of copy library         */
```

Figure 31. Request file to import an IMS/MFS format set

Figure 32 shows the panel and the names of the ISPF variables that you have set with the request file in this example.

```
DGIUE20         SPECIFY IMPORT UTILITY PARAMETERS
Command ===>

Source
  Type  . . . . . .IIVTYP   1. CICS/BMS macros
                            2. IMS/MFS utility statements
                            3. ISPF panel
                            4. GDDM-IMD export data set
                            5. CSP/AD export data set
                            6. SDF/CICS dump data set
                            7. External source format
  Data set name . . . . . IIVNAM
Copy source library (IMS/MFS only)
  Data set name . . . . . IUV20CL
Skeleton panel
  Name  . . . . . . . . . IUV20SD
  Library . . . . . . . . IUV20SL

Target
  Library . . . . . . . . IIVLIB
For existing objects specify
  Option  . . . . IUV20EO  1. Use alternate name automatically
                           2. Specify option by object individually
```

Figure 32. Variables for import utility parameters for IMS/MFS

Note: If an object already exists, SDF II generates an alternative name automatically. You cannot influence this value. You will find the name in the message list.

Example 8: Print an object

Figure 33 is a request file for printing a complete listing of the panel MYPANEL.

To print other objects, specify the appropriate value for IIVTYP. The output is to be formatted for the standard printer.

```
IIVREQ = 'P'              /* requests printing                    */
IIVNAM = 'MYPANEL'        /* identifies the object name           */
IIVLIB = '*'              /* uses search order                    */
IIVTYP = 'P'              /* identifies object as a panel         */
IUV10FMT = '1'            /* formats output for standard printer  */
IUV10CNT = '1'            /* requests complete listing            */
```

Figure 33. Request file to print a panel

Figure 34 shows the panel and the names of the ISPF variables that you have set with the request file in this example.

```
DGIUE10             SPECIFY PRINT UTILITY PARAMETERS
Command ===>

Identify the source object
  Name  . . . . . . . . . IIVNAM
  Library . . . . . . . . IIVLIB
  Type  . . . . . .IIVTYP G. PANEL GROUP           O. CONTROL TABLE
                          P. PANEL                 S. PARTITION SET
                          A. AID TABLE

Output options
  Contents
  IUV10CNT  1. Complete listing
            2. Format only

  Format
  IUV10FMT  1. Standard printer
            2. GDDM Print Utility
            3. Input to DCF
            4. DBCS Printer
            5. DBCS Printer with field outlining
```

Figure 34. Variables for print utility parameters for a panel

Example 9: Print a chapter of the online reference

Figure 35 is a request file for printing the Panel Editor topic of the online reference. The output is to be prepared for the Document Composition Facility (DCF).

```
IIVREQ = '0'              /* requests printing online ref (use letter 0)  */
IIVNAM = '2'              /* prints the panel editor topic                */
IIVMOD = '2'              /* prepares output for DCF                      */
IUV40DSN = "'DCF.INPUT.LIB'" /* identifies the pds output data set        */
```

Figure 35. Request file to print a topic of the online reference

Figure 36 shows the panel and the names of the ISPF variables that you have set with the request file in this example.

```
DGIUE40             PRINT ONLINE REFERENCE PARAMETERS
Command ===>

 Specify online reference topic to be printed
   Print topic . . . IIVNAM
          0 - Complete Online Reference    7 - Generate
          1 - General Information          8 - List Objects
          2 - Panel Editor                 9 - Specify Libraries
          3 - Panel Group Editor          10 - Utilities
          4 - Partition Set Editor        11 - Profile
          5 - Aid Table Editor            12 - System Administration
          6 - Control Table Editor        13 - Prototyping

 Format of the listing
   FORMAT  . . . . . IIVMOD      1 - Standard printer
                                 2 - Input to DCF
                                 3 - DBCS Printer

 Destination for format 2
   DCF DS name . . . IUV40DSN
```

Figure 36. Variables for printing the online reference parameters

Example 10: Construct a panel

Figure 37 is a request file for constructing a panel from predefined elements, such as fields, arrays, or repeat formats. You define these elements in the construction utility table (default name is DGIU5TAB), whose name you specify in the field IIVONAM of the request file. The contents of IIVOLIB are explained in "Using the invocation interface" on page 39.

```
IIVREQ = 'A'               /* requests construction                 */
IIVNAM = 'MYPAN'           /* identifies the object name            */
IIVLIB = '1'               /* stores panel in library 1             */
IIVDEV = '3270'            /* identifies device type                */
IIVONAM ='DGIU5TAB'        /* identifies construction utility table  */
IIVOLIB = '0240800HELPPAN '/* 024 = depth of panel format           */
                           /* 080 = width of panel format           */
                           /* 0 = include a command line            */
                           /* HELPPAN = name of help panel          */
```

Figure 37. Request file to construct a panel

Figure 38 and Figure 39 on page 64 show the panels and the names of the ISPF variables that you have set with the request file and the construction utility table in this example.

```
DGIUE51        SPECIFY PANEL CONSTRUCTION PARAMETERS
Command ===>

Identify the panel
  Name  . . . . . . . . . IIVNAM
  Library . . . . . . . . IIVLIB
  Device type . . . . . . IIVDEV
```

Figure 38. Variables for panel construction parameters

```
DGIUE52                     SPECIFY PANEL ELEMENTS
Command ===>                                                Scroll ===> PAGE

Panel size
  DEPTH . . . . . . . . . (1)           WIDTH . . . . . (1)
Elements  . . . . . . . . . . . . . . . . . . . .  COLUMNS 1-7 OF 7, ROW 0 OF 0
    Related field -- Name ----- Leng Occ Ver Prompt Description ---------------
''' TOP OF DATA ********************************************************************
''' (2)              (3)        (4)  (5) (6) (7)    (8)
'''
'''
'''
'''
'''
'''
'''
''' (2)              (3)        (4)  (5) (6) (7)    (8)
''' END OF DATA ********************************************************************
```

Figure 39. Variables for panel elements for panel construction

The numbers (*n*) in Figure 39 indicate the following variables:

(1) IIVOLIB
(2) IUV5RFD
(3) IUV5NAM
(4) IUV5LEN
(5) IUV5OCC
(6) IUV5VER
(7) IUV5PRO
(8) IUV5DES

These variables are set from your specifications in the corresponding columns of the invocation interface table (default name DGIIFTAB) and the construction utility table (default name DGIU5TAB). See Figure 13 on page 39 and Figure 15 on page 46 for a description of these ISPF tables.

Example 11: Delete an object

Figure 40 is a request file for deleting an object.

```
IIVREQ   = 'D'                  /* REQUEST DELETE           */
IIVNAM   = 'TEST01'             /* OBJECT NAME         INPUT */
IIVLIB   = '2'                  /* LIBRARY NR.         INPUT */
IIVTYP   = 'P'                  /* OBJECT TYPE PANEL        */
```

Figure 40. Request file to delete an object

The panel below shows how the same operation would be done online; note that there are no corresponding variables.

```
DGIOE10                        LIST OBJECTS
Command ===>

Objects . . . . . .
    Name --- Lib Typ Operands   Syst Description ---------------
d'' TEST01   2   P              CICS SAMPLE PANEL 1
```

Figure 41. Corresponding panel

Example 12: Copy an object

Figure 42 is a request file for copying a panel. IIVMOD specifies that an existing panel will be replaced.

```
IIVREQ   = 'C'                  /* REQUEST COPY              */
IIVNAM   = 'TEST01'             /* OBJECT NAME         INPUT */
IIVLIB   = '1'                  /* LIBRARY NR.         INPUT */
IIVTYP   = 'P'                  /* OBJECT TYPE PANEL         */
IIVONAM  = 'TEST01'             /* OBJECT NAME        OUTPUT */
IIVOLIB  = '2'                  /* LIBRARY NR.        OUTPUT */
IIVMOD   = '10'                 /* COPY REPLACE              */
```

Figure 42. Request file to delete an object

The panel below shows how the same operation would be done online; note that there are no corresponding variables.

```
DGIOE10                       LIST OBJECTS
Command ===>

Objects . . . . . .
    Name --- Lib Typ Operands -------------------- Syst Description --
C   TEST01   1   P   TEST01 2 (REP                 CICS SAMPLE PANEL 1
```

Figure 43. Corresponding panel

Example 13: Extract an object

Figure 44 is a request file for extracting a panel.

```
IIVREQ   = 'X'                  /* REQUEST EXTRACTION        */
IIVNAM   = 'TEST01'             /* OBJECT NAME         INPUT */
IIVLIB   = '1'                  /* LIBRARY NR.         INPUT */
IIVTYP   = 'P'                  /* OBJECT TYPE PANEL         */
IUV65UEX = 'DGIUX60C'           /* USER EXIT                 */
```

Figure 44. Request file to extract an object

The figure below shows the corresponding panel.

```
DGIUE65               PANEL DATA EXTRACTION UTILITY
Command ===>
Identify the panel
  Name . . . . . . . . . IIVNAM
  Library  . . . . . . . IIVLIB

Select the formats
  Device type  . . . . . IIVDEV

  User exit  . . . . . . IUV65UEX Performs data extraction
```

Figure 45. Corresponding panel

Example 14: Modify an object

Figure 46 is a request file for modifying a panel.

```
IIVREQ   = 'F'                  /* REQUEST MODIFICATION      */
IIVNAM   = 'TEST01'             /* OBJECT NAME         INPUT */
IIVLIB   = '1'                  /* LIBRARY NR.         INPUT */
IIVTYP   = 'P'                  /* OBJECT TYPE PANEL         */
IIVONAM  = 'TEST01'             /* OBJECT NAME        OUTPUT */
IIVOLIB  = '2'                  /* LIBRARY NR.        OUTPUT */
IUV60ACT = '1'                  /* MODIFY ONLY               */
IUV60UEX = 'DGIUX60C'           /* USER EXIT                 */
```

Figure 46. Request file to modify an object

The figure below shows the corresponding panel.

```
DGIUE60               PANEL MODIFICATION UTILITY
Command ===>
Identify the source object
  Name . . . . . . . . . IIVNAM
  Library  . . . . . . . IIVLIB

Select an action and the formats to which it should be applied
  Action . . . . . . . . IUV60ACT  1. Modify only
                            2. Modify and change device type
                            3. Modify and create copy to new device type
  Device type  . . . . . IIVDEV
  New device type  . . . IUV60ODV      for action 2 and 3
  User exit  . . . . . . IUV60UEX      Performs format modification

Identification of object to be created
  Name . . . . . . . . . IIVONAM
  Library  . . . . . . . IIVOLIB
```

Figure 47. Corresponding panel

Using predefined panel elements

To use predefined panel elements (also called related fields, for instance in the Define Field dialog), you need two user exit routines as interfaces to a database. The user exit routines are:

- Retrieve field (DGIUXRET)

  This user exit routine is used by the panel construction utility or by the Define Fields dialog of the panel editor to retrieve additional information, such as a field's length or occurrence number.

- Expand field (DGIUXEXP)

  This user exit routine is used by the panel construction utility to handle the expand (**ex**) line command and automatic expansion.

Sample routines are available as members DGIUXRET and DGIUXEXP in SDF2.V1R4M0.SDGICMD. These routines are intended to be used with the procedures described in *SDF II General Introduction*. To keep the sample simple, the data to be retrieved is stored in an MVS sequential data set, which you have to create and set up as described in *SDF II General Introduction*. Normally you would store your data in a database rather than in a sequential data set.

Retrieve field

The retrieve field user exit routine is used to bring into SDF II predefined panel elements from, for example, a database.

Input: The following variable is available in the shared pool:

Figure 48. DGIUXRET input variable

Name	Length	Description
IUVUERFD	256	The name of the related field.

- The maximum length of the related field name is 256 characters.
- The name may be qualified.
- The separator is a period.
- The maximum length of a name and each qualifier is 32 characters.
- The name and the qualifiers must not contain blanks.

Invocation: The user exit routine DGIUXRET is called by the standard ISPF invocation:

```
SELECT CMD(DGIUXRET)
```

Output: The user exit routine may set the following variables in the shared pool:

Figure 49. DGIUXRET output variables

Name	Length	Description
IUVUELEN	4	The length of the field (values 0 through 9999).
IUVUEFFO	4	The field format.
IUVUEOCC	3	The occurrence number (values 0 through 999).
IUVUEPRO	20	The field prompt.
IUVUEDES	64	The field description.
IUVUEINI	256	The initial value for the field.
IUVUEMSG	8	The identifier of a private message that SDF II displays if an error is detected (RC=16).

Note: Refer to "The construction utility table" on page 46 for an explanation of the corresponding fields.

Return codes: The return codes from DGIUXRET are:

0 The related field was successfully retrieved.
16 The related field was not retrieved. The reason is given in the message whose identifier is specified in IUVUEMSG.

SDF II issues message number DGIUM571 for RC=16. To issue a private message instead:

- Assign the number of the private message to the variable IUVUEMSG in the DGIUXRET EXEC.
- Define the message in ISPF message format.
- Use DGIIXIN1 to add an appropriate LIBDEF statement for ISPMLIB to the invocation routine SDF2INV.

Expand field

For a field for which the expand (**ex**) line command is entered or for which an automatic expansion is performed, the user exit routine DGIUXEXP is called.

SDF II performs an automatic expansion if you use the invocation interface to invoke the panel construction utility. A panel element is expanded to the next level provided there is one and provided the resulting panel elements are correct. This process is repeated until no further expansion is possible because the lowest level is reached.

Input: The following variable is available in the shared pool:

Figure 50. DGIUXEXP input variable

Name	Length	Description
IUVUERFD	256	The name of the related field.

The maximum length of the related field name is 256 characters. The name may be qualified. The separator is a period. The maximum length of a name and each qualifier is 32 characters. The name and the qualifiers must not contain blanks.

Invocation: The user exit routine DGIUXEXP is called by the standard ISPF invocation:

```
SELECT CMD(DGIUXEXP)
```

Output: The user exit routine creates the ISPF table DGIUXEXP, which may be permanent or temporary. One row is added each time the field is expanded. The columns of DGIUXEXP are:

Figure 51. DGIUXEXP output variables

Name	Length	Description
IUVUNAM	32	The SDF II field name.
IUVUELEN	4	The length of the field (values 0 through 9999).
IUVUEFFO	4	The field format.
IUVUEOCC	3	The occurrence number (values 0 through 999).
IUVUEPRO	20	The field prompt.
IUVUEDES	64	The field description.
IUVUEINI	256	The initial value of the field.
IUVUEMSG	8	The identifier of a private message that SDF II displays if an error is detected (RC=8 or RC=16).

Note: Refer to "The construction utility table" on page 46 for an explanation of the corresponding fields.

Return codes: The return codes from DGIUXEXP are:

0 The related field was successfully expanded.
8 Expansion is not possible because the last level of the field was reached.
16 The last level was not reached, and the related field was not expanded. The reason is given in the message whose identifier is specified in IUVUEMSG.

SDF II issues message number DGIUM571 for RC=16 and message number DGIUM572 for RC=8. To issue private messages instead:

- Assign the number of the respective private message (RC=8 or RC=16) to the variable IUVUEMSG in the DGIUXEXP EXEC.
- Define the messages in ISPF message format.
- Using DGIIXIN1, add an appropriate LIBDEF statement for ISPMLIB to the invocation routine SDF2INV.

Using generated output

This section describes the types of generated output and how to further process them. The generated output depends on the target that you have specified for the generation on the Identify Object for Generation panel.

Prototype: The generated output is an ISPF panel with additional information for application prototyping. Use it in the SDF II prototype facility to define or simulate a prototype.

See “Defining prototype libraries” on page 16 for what you have to do before you can use the SDF II prototype facility.

CICS/BMS: Depending on what you specified on the Specify Generation Parameters panel, the generated output consists of either or both:

- CICS/BMS macros, created from SDF II panels, panel groups, and partition sets. Process them in the same way that you would process macro statements that were not created by SDF II, but, for example, with a standard editor.
- Application data structures. Include them into your source program.

IMS/MFS: Depending on what you specified on the Specify Generation Parameters panel, the generated output consists of either or both:

- IMS/MFS utility control statements, created from SDF II panels, AID tables, partition sets, and control tables. Process them in the same way that you would process IMS/MFS format sets that were not created by SDF II, but, for example, with an editor.
- Application data structures. Include them into your source program.

ISPF: Depending on what you specified on the Specify Generation Parameters panel, the generated output consists of either or both:

- An ISPF panel. Put this panel directly into the panel library of your application program.
- Application data structures. Include them into your source program.

  Besides the data structures for your application program, data structures are generated for use within ISPF VDEFINE and VDELETE service calls. You must code the ISPF service calls yourself. Skeletons are provided as examples of these calls.

  The generated data structures represent the parameters to the VDEFINE/VDELETE service calls for a panel. It is assumed that all variables of a panel are defined by a single service call with the LIST option.

  The data structures can also be used for ISPF Version 4 Release 1. Routines for use in this case are provided in the sample library.

  The sample library, SDF2.V1R4M0.SDGISAM contains the following members:

DGICSC3V	Service call examples for language COBOL and ISPF Version 2 Release 3 or later
DGICSP3V	Service call examples for language PL/I and ISPF Version 2 Release 3 or later
DGICSX3V	Service call examples for language C and ISPF Version 3 or later
DGICSC2R	Routine example for language COBOL and ISPF Version 2 Release 2
DGICSC2V	Service call examples for language COBOL and ISPF Version 2 Release 2
DGICSP2D	Entry point declaration example for language PL/I and ISPF Version 2 Release 2
DGICSP2R	Routine example for language PL/I and ISPF Version 2 Release 2
DGICSP2V	Service call example for language PL/I and ISPF Version 2 Release 2
DGICSA2R	Routine example for language Assembler and ISPF Version 2 Release 2.

GDDM-IMD: The generated output is a GDDM-IMD export data set, created from SDF II panels, a panel group, and AID tables. Use the GDDM-IMD import utility to import the export data set into the MSL.

CSP/AD: Depending on what you specified on the Specify Generation Parameters panel, the generated output is either:

- CSP/AD external source format

  Use the import utility of CSP/AD to import the external source format into the MSL.

- CSP/AD export data set, created from SDF II panels or panel groups, or both.

  To process export data sets, follow these steps:

  1. Use the GET option of the ALF utility to read the export data set. The ALF utility is a CSP application named UTILALF.ALFUTIL that runs under CSP/AE.
  2. Use the import utility of CSP/AD to import the map or map set from the ALF data set into the MSL.

  SDF II provides an example user exit routine to perform these tasks for the CSP/AD generation. It is available as member DGICXCSP in SDF2.V1R4M0.SDGISAM. Adapt this EXEC to suit your installation.

Writing a generation user exit routine

When you generate an object, you can specify the name of a user exit routine to be invoked after successful generation.

For CICS/BMS, IMS/MFS, or ISPF, you can use two different user exit routines: one for application data structure generation, the other for CICS/BMS macro generation, IMS/MFS utility control statement generation, or ISPF panel generation. For CICS/BMS panels, the user exit routine for CICS/BMS macro generation is invoked after generation for each format instance.

For GDDM-IMD and CSP/AD, you have only one user exit routine.

All user exit routines are REXX EXECs. They can access a number of variables stored in the ISPF shared pool. They can be used to further process the generated output. For example, they could submit a job to assemble and link edit the generated CICS/BMS macros, or a job to process the generated IMS/MFS utility control statements with the MFS language utility.

The following variables are available to the user exit routine:

Figure 52 (Page 1 of 2). generation user exit routine

Column	Length	Description
ICVUENAM	8	The name of the object that has been generated.
ICVUETYP	1	The type of object that has been generated. The following values are possible: G Panel group P Panel S Partition set
ICVUELIB	1	The identifier of the library in which the object that has been generated is stored.

Figure 52 (Page 2 of 2). generation user exit routine

Column	Length	Description
ICVUETSY	1	The target system for which generation is done. The following values are possible: 0 Generation for application prototype 1 Generation for CICS/BMS 2 Generation for IMS/MFS 3 Generation for ISPF 4 Generation for GDDM-IMD 5 Generation for CSP/AD
ICVUEGTY	1	The type of generation. The following values are possible: D A data structure has been generated. C CICS/BMS macros, IMS/MFS utility control statements, ISPF panels, or a GDDM-IMD or CSP/AD export data set have been generated.
ICVUELAN	1	The programming language of the generated data structure. The contents of this variable are undefined when the value of ICVUEGTY is C, or when generating for a target system other than CICS/BMS, IMS/MFS, or ISPF. The following values are possible: A Assembler C COBOL P PL/I R RPG II (this is supported only for CICS/BMS) X C language
ICVUEODD	8	The ddname used for the output file.
ICVUEOLN	46	The data set name of the output file.
ICVUEOMN	8	The member name of the file containing the generated output.

Note: ICVUEODD and ICVUEOLN are mutually exclusive. If ICVUEODD is blank, ICVUEOLN is defined, and vice versa.

An example of a user exit routine that shows the generated output by using the PDF browse services is available as member DGICXBRW in SDF2.V1R4M0.SDGISAM.

Note: If you use ISPF services, such as BROWSE or EDIT, in the generation exit routine, set the variable DGISCACT to blank before you invoke the ISPF services. Otherwise, you will not be able to scroll the panel.

Getting the bill of material table

SDF II creates a table (DGICVBOM) that contains all objects that were referenced during generation.

The following variables are available:

Figure 53 (Page 1 of 2). bill of material table

Column	Length	Description
IIVBOMN	≤8	The name of the referenced object.

Figure 53 (Page 2 of 2). bill of material table

Column	Length	Description
IIVBOMT	6	The type of the referenced object. The following values are possible: DGIPNL For a panel DGIGRP For a panel group DGIPST For a partition set DGITBL For an AID table DGIOCT For a control table
IIVBOML	1	The identifier of the library in which the object is stored. Values 1 through 9 are possible.

Chapter 5. Customizing

This chapter explains system programmer's tasks, such as:

- Defining defaults, device types, CUA elements, and emphasis classes
- Translating text used in SDF II
- Adapting print utility statements
- Changing the uppercase/lowercase translation table

For most of these customization tasks, you can use SDF II dialogs. For full information refer to the online reference.

Making changes available to other users

You can customize various parts of SDF II, as described in this chapter. Any parts you change are put into the ISPF table output library (ddname ISPTABL) as members with names in the format DGIK*xxxx* or the member DGIPROF in the ISPF profile (ISPPROF). If you have not changed the corresponding statement in the invocation routine, your ISPF profile will be used as your ISPF table output library.

Allocate the data set that you use as table output library as the first data set in your table input library (ddname ISPTLIB). This ensures that your customization will be effective for testing.

To make the changed parts available to other users, move these files to the table input library SDF2.V1R4M0.SDGITB*xx*. To do this, use the ISPF move utility (option **3.3**).

If you have customized the profile DGIPROF, delete this member from the users' profile library.

Note: This deletes any profile customization an individual user has done.

After making the changed tables available to other users, delete the copies contained in your table output library. Otherwise, you will be using the tables from the table output library instead of the ones made generally available.

This is particularly important if you or someone else applies maintenance affecting the tables. After any such maintenance, you may need to customize the tables again. All tables are members whose names either begin with DGIK, or it is the member DGIPROF. These members should be deleted from your private table library once they are generally available.

Notes:

1. Always keep copies of the original tables in case you want to use the original SDF II defaults.
2. Create a backup copy of the customized tables because any program maintenance may change them.

Defining standard defaults

To define the standard defaults for SDF II, enter the **sdf2** command and select option **10** on the Select an SDF II Function panel. This panel is then displayed:

```
 Exit  Help
--------------------------------------------------------------------------------
                        SELECT A PROFILE EDITOR DIALOG
Option ===>
1   SYSTEM ENVIRONMENT     Specify target system

2   DEFAULTS               Specify overall editing defaults

3   DIALOGS                Customize SDF II windows

4   PRINTER                Specify print page size

5   ISPF PARMS             Specify terminal and user parameters
```

Specify target system

In this dialog, you define the target system. The target system can be CICS/BMS, IMS/MFS, ISPF, GDDM-IMD, or CSP/AD.

Specify overall editing defaults

In this dialog, you can specify various defaults.

1. Defaults for all SDF II editors:
   - **Autosave** specifies whether and how often an edited object is to be saved automatically in the autosave library.
   - **Numbers** turns on or off the line numbers of window panels.
2. Defaults for the panel editor:
   - **Preserve** defines whether the data structure is protected during editing.
   - **Capitals** controls the translation of lowercase letters to uppercase letters in the Format window.
   - **Nulls** controls whether trailing nulls or blanks fill each line of the Format window of the panel editor.
   - **Linecommands** controls whether the line command area is to be shown in the Format window.
   - **Mixed-case DSECT names** defines whether data structure names can be specified in mixed case.
   - **Marks** defines the characters to be used as marks when creating a new panel.
3. Confirmation of line deletion requests:

   Confirmation defines whether the users are asked to confirm a **delete** line command in the List Objects and Define Fields dialog.

Customize SDF II windows

Use the Customize a Window panel to change the layout of the selected window. You can choose the columns displayed in the dialog, and the width of each column. Changes in the layout of the window apply only to the target system defined in the profile editor. You can define different layouts of the same window for each target system.

Specify print page size

In this dialog you define the page size used by the print utility.

If you change the page width and want to print complete objects (not only the formats), you need to adjust the skeletons that define the print layout. For details on how to do this, see “Adapting print utility skeletons” on page 91.

Take care that the record sizes of the data sets used to store the output of the print utility correspond to the page widths.

Specify terminal and user parameters

In this dialog you can display and modify selected ISPF parameters. For more information, see the ISPF manuals.

Making changed defaults available

When you have finished defining the defaults and leave the dialog, you get this member in your ISPF profile library:

DGIPROF

How to give other users access to this table is explained in “Making changes available to other users” on page 77.

Specifying libraries

In the Specify Libraries dialog, you specify the libraries you want to use in SDF II (option 8).

When you have finished specifying you libraries and leave the dialog, you get this member in your ISPF profile library:

DGIPROF

How to give other users access to this table is explained in “Making changes available to other users” on page 77.

Defining device types

The device table distributed with SDF II lists all the types of devices supported by SDF/CICS, CICS/BMS, IMS/MFS, GDDM-IMD, and CSP/AD. In addition, you can define new device types or device lists that are suitable for your site.

You can enter the SDF II Device Type Editor by means of the **sdf2c** command, which displays the customization dialogs.

In SDF II, device types can be names of devices or names of device lists. A device name corresponds to a single type of device supported by the target systems. A device list contains the names of a set of devices that have the same characteristics.

The SDF II functions do not distinguish between devices and device lists. Wherever you are required to enter the name of a device type, you can enter the name of either a device or a device list.

When necessary (such as for generation), each device in the device list will be processed separately. Generation, for example, is then performed for each device in the list.

Devices

On the Select SDF II Customization Dialog panel, select option **1** to display the Define Devices panel, where you can find and specify:

- The device name
- An indication as to whether this is a device list
- Entries that show the corresponding device code in the target systems
- A column with a comment.

These line commands are available on the Define Devices panel:

c Copy device type entries
d Delete device type entries
i Insert new devices
s Select and display:
  - The Define Device List panel for a device list. There you define the devices that belong to the device list.
  - The Define Device Type Characteristics panel for single devices. There you define the characteristics of the device. You cannot change the characteristics of the devices in the list provided by IBM.

Make sure that the definition of the device type contains an entry for each target system for which the device type may be used. Otherwise you cannot create or generate an object for this device and target system.

The import utility uses the first device type that matches the device characteristics of the object to be imported as the device type of the imported object. Consider this matching algorithm when you add or change entries in the device table.

- The CICS/BMS device suffixes for target systems in SDF II may differ from the device suffixes to be imported. If this is the case, an entry specifying the correct suffix needs to be added to the device table to import the object successfully. Suffixes of device entries provided by IBM cannot be changed.
- If you used values other than the default values in the device characteristics table (DVDDCT) of SDF/CICS or in the IMS system generation, modify the SDF/CICS device code (column SDF) and the IMS/MFS device type (column MFS) in the device table.
- Because the import utility takes the first matching entry in the device table, the resulting device type may be different in size or supported attributes from the size or attributes of the object to be imported.

To avoid the loss of attributes, you may need to replace the entry in the target system column with blanks (for IMS/MFS and SDF/CICS this column can be modified) or to add your own device type to the device table before the first matching device type provided by IBM.

Device list

A device list contains a set of devices that have the same characteristics. It cannot contain another device list. Define a device in the SDF II device table, then include it in a device list. A device list can contain up to 24 devices.

The device characteristics of a device list are the superset of the characteristics of the devices in the list. But all device types contained in the device list must have the same setting for the device attribute `ADJACENT FIELDS`.

A device list for IMS/MFS should contain only devices of the IBM 3270 display family or the IBM 3270 printer devices.

Note: It is recommended to group only devices with the same display size or page size into one device list.

Device type characteristics

On the Define Device Type Characteristics panel you can define the following characteristics:

- Device dimensions: depth, width, and an indication as to whether the device size can be extended
- Character cell size (vertical and horizontal)
- Information about partitioning
- Device attributes (adjacent fields, numeric lock feature, light pen, number of PF keys)
- Input, output
- Extended device attributes (such as extended colors and extended highlighting)
- Device features (such as tabulators).

Making changed device types available

When you have finished defining your own device table and leave the dialog, you get these two tables in your ISPF table output library:

DGIKFDT
DGIKF10

How to give other users access to these tables is explained in “Making changes available to other users” on page 77.

Translating

You can change all text that is displayed by the SDF II program:

- Messages
- Online reference
- Help panels
- Function panels
- Operator input
- Column headings used in the SDF II windows

- Panel commands and parameters
- Line commands
- Panel text used in the SDF II windows
- Print utility skeletons.

For skeletons and messages, use a standard text editor. For panels you can use the SDF II panel editor or a standard text editor. For all other objects use the SDF II translation dialogs.

Making translated parts available to other users

To make translated parts available follow these steps:

1. Put the translated files into partitioned data sets. Use file names that are different from the file names of the following distributed data sets:

SDF2.V1R4M0.SDGIMS*xx*	For the message library
SDF2.V1R4M0.SDGITU*xx*	For the online reference panel library
SDF2.V1R4M0.SDGIHP*xx*	For the help panel library
SDF2.V1R4M0.SDGIPN*xx*	For the function panel library
SDF2.V1R4M0.SDGITB*xx*	For the table library, which contains the tables for: Operator input Column headings Panel commands Line commands Panel text Emphasis class CUA attributes
SDF2.V1R4M0.SDGISK*xx*	For the skeleton library.

   xx in the data set names is, according to the installed language,

   DU for German,
   JN for Japanese,
   EP for Spanish,
   DS for Swiss German, or
   EU for US English.

2. Adapt the assignment statements that define the libraries in the invocation routine SDF2INV via the DGIIXIN1 routine.

   For example, if you have translated

   - Help panels, which are now in SDF2.MY.DGIHLIB,
   - Messages, which are now in SDF2.MY.DGIMLIB,
   - Line commands and panel commands, which are now in SDF2.MY.DGITLIB,

   use DGIIXIN1 to customize the dataset name prefix for SDF II data sets. as described in “General definitions” on page 6.

Notes:

1. Because of possible program service, it is better not to replace members in the original data sets.
2. If you intend to use the print online reference utility of SDF II, the allocation for DGIULIB must not contain more than four libraries.

Messages

The messages are in the partitioned data set SDF2.V1R4M0.SDGIMS*xx*. The syntax of the member names is:

DGI*f*M*nn*

Where:

f is the function code
nn is a number.

The SDF II messages are in ISPF message format. Each message consists of two lines.

Here is, for example, the member DGILM33:

```
.CCSID=00037
DGILM331 'Disk is full            ' .HELP=DGILH331  .TYPE=A
'Clear some disk space or specify another disk before continuing     '
DGILM332 'Permanent I/O error     ' .HELP=DGILH332  .TYPE=A
'&IYVMINS2  RC=3; permanent I/O error, please re-access disk         '
DGILM333 'Error while writing     ' .HELP=DGILH333  .TYPE=A
'FSWRITE  RC=&IYVMINS3; unable to continue writing to disk           '
DGILM334 'No output               ' .HELP=DGILH334  .TYPE=A
'No records have been written to the ouput file.                     '
DGILM335 '                        ' .HELP=DGILH335  .TYPE=A
'Output error on &IYVMINS4                                           '
```

Figure 54. Message example (DGILM33)

Do not change the message inserts (&IYVMINS*n*) or variables of either the long message or the short message. When you translate, take the length of message inserts and variables into account.

On the first message line, translate the short message. The short message can be up to 24 characters long after variables have been substituted with text.

On the second message line, translate the long message. The long message can be up to 78 characters long after variables have been substituted with text.

Online reference

The online reference panels are in the partitioned data set SDF2.V1R4M0.SDGITU*xx*. The syntax of the member names is:

DGI*f*U*xxx*

Where:

f is the function code
xxx is an alphanumeric string.

The SDF II online reference is in ISPF tutorial format.

Figure 55 on page 84 shows an example of the member DGIUU300.

```
)CCSID NUMBER(00037)
)PANEL KEYLIST(DGIYYKOR,DGI)
)ATTR DEFAULT(   )
   TYPE(PT)
   TYPE(NT)
   TYPE(TEXT) INTENS(NON)
   TYPE(ET)
   TYPE(FP)
 TYPE(PS)
   TYPE(CH)
   TYPE(NEF) CAPS(ON)
   TYPE(SAC)
 ┼ AREA(SCRL) EXTEND(ON)
 ┴ TYPE(PIN)
)BODY
                              Conversion Utility
 Command ===> ZCMD

┼area1                                                                    ┼
)AREA AREA1
    Select the conversion utility to convert objects from one target system
    to a new target system. The converted objects replace the source
    objects.
 IUVIDX 004
    Any features not supported by the new target system are deleted on
    conversion. Features required by the new target system, but not defined,
    are assigned a default value during conversion. Upon conversion, the
    object is modified to comply with the rules of the new target system.

    For more information, see the following topics:
 IUVSEL
┴For more information, see the following topics:

     1Entry from Utility Selection
     2Entry from List Objects
     3Conversion Rules
)INIT
 &IUVIDX   = &Z
 /* ))DGI INDEX: conversion utility */
 /* ))DGI INDEX: utility, conversion */
 /* ))DGI INDEX: object, converting */
 /* ))DGI INDEX: converting SDF II objects */
)PROC
 &ZUP      = DGIUU000
 &ZSEL     = TRANS(&ZCMD       /* select next topic */
                   1,DGIUU310
                   2,DGIUU320
                   3,DGIUU330
                   )
)PNTS
 FIELD(ZPS01001) VAR(ZCMD) VAL(1)
 FIELD(ZPS01002) VAR(ZCMD) VAL(2)
 FIELD(ZPS01003) VAR(ZCMD) VAL(3)
)END
```

Figure 55. Example of an online reference panel (DGIUU300)

Note: Be aware that the panels use special characters that might not be displayed on your terminal.

Do not change the)ATTR section of the online reference panels, except when one of your national language characters is being used as an attribute character. In this case, change that attribute character in both the)ATTR and the)BODY sections.

Translate the text of the)BODY section. Here are some guidelines:

- The depth must be 24 lines.
- The width must be 80 characters.
- Every line must start with an attribute character.
- DBCS fields and SO/SI pairs must not wrap around lines.
- Do not change the attribute positions in the first line (header line).
- Line 3 must be blank. It is reserved for system messages.
- Line 23 must be blank. It is reserved for system information.
- Line 24 contains instructions on how to proceed.
- ISPF variables in the form &IKK*xxxxx* are keywords, defined in the translation dialogs. They are used to make the translation available globally. Do not change them here.
- The variable &IUVSEL separates text from the selection menu on a selection panel. Everything that follows this variable is not included in the printed version of the online reference.
- Do not use expansion marks.

In the)INIT section, translate the *text* that follows any /*))DGI INDEX:

```
/* ))DGI INDEX: text     */
```

This *text* goes into the index of the printed reference information. It cannot be longer than 30 characters. The online index must be coded manually.

Help panels

The help panels are in the partitioned data set SDF2.V1R4M0.SDGIHP*xx*. The syntax of the member names is:

DGI*f*H*xxx* message panels

DGI*f*I*xxx* field help panels

Where:

f is the function code
xxx is an alphanumeric string.

Figure 56 on page 86 shows an example of the member DGIOI104.

```
)CCSID NUMBER(00037)
)PANEL KEYLIST(DGIYYKFH,DGI)
)ATTR DEFAULT(   )
   TYPE(PT)
   TYPE(NT)
   TYPE(ET)
   TYPE(CH)
   TYPE(OUTPUT) COLOR(WHITE) INTENS(HIGH) SKIP(ON)
 ┼ AREA(SCRL) EXTEND(ON)
)BODY WINDOW(75,14) CMD() LMSG(LMSG)
                   Library Identifier Search Argument
 LMSG
┼area1                                                                  ┼
)AREA AREA1
 Specify the library that SDF II is to search for objects or that all
 specified libraries are to be searched.

 Possible values:

    1-9    A library identifier
    blank  to include objects from all libraries.

)INIT
 &IUVIDX   = &Z
)PROC
 &ZUP      = DGIOU100
)END
```

Figure 56. Example of a help panel (DGIOI104)

Note: Be aware that the panels use special characters that might not be displayed on your terminal.

When you translate the help panels, follow the translation guidelines in "Online reference" on page 83.

Function panels

To translate the function panels, use the SDF II panel editor or a standard text editor. You can get basic information on how to use SDF II by reading *Designing Panels with SDF II*, SH19-8212-0.

Before you use the SDF II panel editor to translate panels, use the import utility to import them to SDF II. You can find further information about how to import ISPF panels in "Importing objects from ISPF" on page 119.

The function panels are in the partitioned data set SDF2.V1R4M0.SDGIPN*xx*. The syntax of the member names is:

DGI*f*E*xxx*

Where:

f	is the function code
xxx	is an alphanumeric string.

After you have imported the function panels to SDF II, set PRESERVE to PERMANENT in the SDF II Profile Editor. The application preserving mode protects the data structure.

To translate text, enter the panel editor and select the Define Format dialog. Check the mark characters first and, if necessary, change those characters that are part of your national language alphabet to other characters that would not otherwise be used. Here are some guidelines:

- Do not modify field attributes.
- Do not modify the sequence of variables.
- Do not modify ISPF variables (starting with &).

Use the **test** command in the SDF II panel editor to check the layout of the translated panel.

When you have finished translating the panels, use the SDF II generation utility to generate the ISPF panels.

Operator input

Use the translation dialog that SDF II provides to translate operator input. To get to the translation dialog enter **sdf2c** option **2.1**.

Type the translated value into the TRANSLATED column. Here are some guidelines:

- The MAXLEN column shows the maximum length of the translated value.
- Make sure that your translated values are unique within one group identified by the number in the GRP column.
- You cannot translate into DBCS or mixed DBCS.
- If you omit the minimum length entry, SDF II automatically calculates a correct value. You can change the minimum length, but take care to avoid ambiguity.

Making changed operator input tables available

When you have finished defining your own operator input table and leave the dialog, you get these two tables in your ISPF table output library:

DGIKFOI
DGIKF21

How to give other users access to these tables is explained in “Making changes available to other users” on page 77.

Column headings

Use the translation dialog that SDF II provides to translate column headings. To get to the translation dialog enter **sdf2c** option **2.2**.

If you want to define mixed DBCS column headings, enter **sdf2c** option **2.2d**.

Type the translated value into the TRANSLATED HEADER column. The maximum length is 24 characters.

Making changed column headings available

When you have finished defining your own column headings and leave the dialog, you get these two tables in your ISPF table output library:

DGIKFCH
DGIKF22

If you use option **2d**, the tables are:

DGIKFCHJ
DGIKF22J

How to give other users access to these tables is explained in "Making changes available to other users" on page 77.

Panel commands and parameters

Use the translation dialog that SDF II provides to translate panel commands and their parameters. To get to the translation dialog enter **sdf2c** option **2.3**.

Type the translated value into the TRANSLATED column. Here are some guidelines:

- The maximum length is 16 characters.
- Make sure that your translated values are unique within one group identified by the number in the GRP column.
- You cannot translate into DBCS or mixed DBCS.
- If you omit the minimum length entry, SDF II automatically calculates a correct value. You can change the minimum length, but take care to avoid ambiguity. Enter **f** if you want the minimum length to be identical to the length of the translated command.

Making changed panel commands available

When you have finished defining your own panel commands and leave the dialog, you get these two tables in your ISPF table output library:

DGIKFPC
DGIKF23

How to give other users access to these tables is explained in "Making changes available to other users" on page 77.

Line commands

Use the translation dialog that SDF II provides to translate line commands. To get to the translation dialog enter **sdf2c** option **2.4**.

Type the translated value into the TRANSLATED column. Here are some guidelines:

- The maximum length is 3 characters.
- Make sure that your translated values are unique within one group identified by the number in the GRP column.
- You cannot translate into DBCS or mixed DBCS.

Making changed line commands available

When you have finished defining your own line commands and leave the dialog, you get these two tables in your ISPF table output library:

DGIKFLC
DGIKF24

How to give other users access to these tables is explained in “Making changes available to other users” on page 77.

Panel text

Use the translation dialog that SDF II provides to translate panel text. To get to the translation dialog enter **sdf2c** option **2.5**.

If you want to define mixed DBCS panel text, enter **sdf2c** option **2.5d**.

Type the translated value into the TRANSLATED VERSION column under the English language version. The maximum length for the translated version is shown in the MXL column.

Making changed panel texts available

When you have finished defining your own panel text and leave the dialog, you get these two tables in your ISPF table output library:

DGIKFTX
DGIKF25

If you used option **5d**, the tables are:

DGIKFTXJ
DGIKF25J

How to give other users access to these tables is explained in “Making changes available to other users” on page 77.

Print utility skeletons

The print utility skeletons that can be translated are in the partitioned data set SDF2.V1R4M0.SDGISK*xx*. They are in ISPF skeleton format and are used to build a temporary file in the first pass of the print utility. A comment box at the beginning of each skeleton tells you whether translation is necessary.

Here are some guidelines that apply to the translation of skeleton members of the form DGIUS1*xx*. These skeletons are used in the print utility.

- Translate all lines whose first character is *. Leave all other lines unchanged.
- Leave items that begin with an & unchanged and in their original position.
- Look at the function panels that are named in the comment at the beginning of the skeleton. Use the same wording as on the translated function panels.
- You can insert DBCS strings by using SO/SI sequences, but make sure they do not wrap lines. You cannot use field outlining.

Here are some guidelines that apply to the translation of skeleton members of the form DGIUS4*xx*. These skeletons are used for printing the online reference.

- Translate all lines whose first character is *.
- Do not change GML tags.
- You can insert DBCS strings by using SO/SI sequences, but make sure they do not wrap around lines. You cannot use field outlining.
- When you have finished translating the print utility skeletons, enter the Print Online Reference dialog. Then print and verify the complete reference for all output formats.

Defining emphasis classes

You can combine presentation attributes into groups called emphasis classes. Each emphasis class is identified by a 2-byte name.

Emphasis classes are a convenient means to change the attributes of screen fields throughout the installation, without having to re-edit the panels: you need only regenerate them.

You define emphasis classes in the customization dialogs. Enter the **sdf2c** command to start the customization dialogs. On the Select SDF II Customization Dialog panel, select option **3** to display the Define Emphasis Class panel. That panel lists the previously defined emphasis classes. On it you can specify:

- The name of the emphasis class
- The list of attributes included
- A column with a comment.

You can create, delete, and modify emphasis classes by using the following line commands, which are available on the Define Emphasis Class panel:

c Copy emphasis class entries
d Delete emphasis class entries
i Insert new entries
r Repeat an entry.

You cannot delete or rename the emphasis classes provided by IBM. The SDF II panel construction utility uses them.

To use an emphasis class, specify the 2-byte name in the panel editor instead of specifying single attributes. The correct syntax is: `CLASS cc`, where *cc* is the name of the emphasis class.

You can define emphasis classes for fields, marks, and attribute descriptors.

When SDF II generates or tests a panel, it replaces any defined emphasis classes with the attributes they represent.

Making changed emphasis classes available

When you have finished defining your emphasis classes and leave the dialog, you get this table in your ISPF table output library:

DGIKF30

How to give other users access to this table is explained in “Making changes available to other users” on page 77.

Specifying CUA panel element attributes

The CUA panel element types are predefined in SDF II. You can, however, change the attributes and the comment for each CUA panel element type.

To make these changes, enter the **sdf2c** command to start the customization dialogs. The Select SDF II Customization Dialog panel is displayed.

Change attributes

On the Select SDF II Customization Dialog panel, select option **4** to display the Specify CUA Attributes panel. This panel lists the CUA panel element types with their previously specified attributes and comments.

The protection attribute is displayed, but cannot be changed.

On this panel you can specify or change the attributes for each CUA panel element type.

You can make any line the first line by using the **/** line command. No other line commands are available on the Specify CUA Element Attributes panel.

When SDF II generates a panel for ISPF, it generates the corresponding CUA attributes. For all other target systems, it replaces any defined CUA attributes with the attributes they represent.

Comments

On the Select SDF II Customization Dialog panel, select option **2** to display the Select a Translation Dialog panel. There, select option **5**, Translate Panel Text, and all text that can be translated is displayed.

Page forward until the CUA panel element types and the associated comments are displayed. You can change the comments by overtyping the text in the `TRANSLATED VERSION` column.

Making changed attributes and comments available

When you have finished specifying attributes and changing comments and leave the dialog, you get this table in your ISPF table output library:

DGIKF40

How to give other users access to this table is explained in “Making changes available to other users” on page 77.

Adapting print utility skeletons

General-use programming interface

The print utility of SDF II uses skeletons to produce the printed output. These print utility skeletons are in the partitioned data set SDF2.V1R4M0.SDGISK or SDF2.V1R4M0.SDGISK*xx*.

During the first pass of the print utility, ISPF file tailoring services are used to create a temporary file, the print input file, from the appropriate print utility skeletons. Variables and tables are replaced by their actual contents. In the second pass of the print utility, SDF II uses this print input file to create the printed output.

Why to adapt

You may need to adapt print utility skeletons because:

- Specifications for DCF in your applications may differ from the defaults. In skeleton DGIUS10D you can change:
  - The DCF continuation character (the default is +)
  - The DBCS replacement string (the default is **)
    For the Japanese language support, the default is no replacement.
  - The string that is inserted at the beginning of each panel line (the default is one blank character)
  - The string that is inserted at the end of each panel line (the default is a null string)
  - Strings that are inserted at the beginning or end, or both, of a specific attribute combination. The defaults are:

bright	:hp2. :ehp2.
underscore	:hp1. :ehp1.
red	:hp1. :ehp1.

  - The new-line character, which allows you to insert multiple DCF control words in one line and forces them to be printed as multiple lines (the default is #).

  In the Japanese NLS feature, this skeleton is changed to allow output to be printed by the DCF/Double Byte licensed program.
- You want to print the message list on a different printer.

  Change the general control record in DGIYSYMS (see “General control record” on page 94). For example, if you change it to `G &IYVMSPAG 120 60 4`, the output will be routed to a DBCS printer.
- DCF control words need to be changed or added to conform to your applications.

  You can do this in those print utility skeletons that contain sections for DCF output. These sections start with `)SEL &IUV14FMT = 3` and end with the corresponding `)ENDSEL`.

  If you add lines with DCF control words to sections that do not yet contain coding for DCF, enclose the added lines in `)SEL` and `)ENDSEL` statements. By doing this, you avoid printing DCF control words for non-DCF output.

  Note that pairs of `)SEL and )ENDSEL` statements can be nested.
- The print width of your printer is different from the default width. The print utility skeletons are prepared for a print width of 120 characters.

  Because the record length of the print input file is 80 bytes, you may need continuation lines. A logical line (variables replaced by their values and continuation lines concatenated) must not exceed the defined width. If any lines exceed

the defined width, they are truncated and you get an error message after printing.

Before you change the print width in the print utility skeletons, you must change the print width using the profile editor.

Formats are adapted automatically to a different print width. You do not need to change the skeletons.

If you want to print other parts of an object, you have to adapt the skeletons.

Note: Always keep copies of the original print utility skeletons in case you want to use the original SDF II settings.

Where to adapt

When the SDF II print utility creates the print input file, it includes print utility skeletons appropriate for the kind of print request.

Figure 57 explains, where to find the information that you may need to adapt to your requirements.

Figure 57 (Page 1 of 2). Skeleton members

For printing ...	You can change ...	In skeleton ...
Any object	General control records General header settings	DGIUS1IG
Any object for DCF output	DCF control records	DGIUS10D
Panel groups	Panel list	DGIUS1G1
	Panel group characteristics	DGIUS1G2
	BMS characteristics	DGIUS1G3
	Programmed symbol set list	DGIUS1G4
	Layout	DGIUS1G5
Panels	Format	DGIUS1P1
	General characteristics	DGIUS1P2
	GDDM characteristics	DGIUS1P3
	BMS characteristics	DGIUS1P4
	Action bar choices	DGIUS1PA
	Fields	DGIUS1P5
	Attributes	DGIUS1P6
	ISPF characteristics	DGIUS1P7
	CSP characteristics	DGIUS1P8
	Editing characteristics	DGIUS1P9
	Structure	DGIUS1PB
	MFS format set	DGIUS1PC
	MFS characteristics / structure	DGIUS1PX
	Format for MFS stream devices	DGIUS1PZ
AID table	AID table	DGIUS1A1

Figure 57 (Page 2 of 2). Skeleton members

For printing ...	You can change ...	In skeleton ...
Control table	Control table	DGIUS1B1
Partition sets	Partition set characteristics	DGIUS1S1
	Partition list	DGIUS1S2
	Layout	DGIUS1S3
Prototype table	Prototype table	DGIUS1W1
Message list	General controls	DGIYSMS

What to adapt

In the print utility skeletons, you can change lines or add new lines if necessary. The lines of the print utility skeletons are:

- **File tailoring statements** are the lines that start with a right parenthesis [)].
- **Print utility records**, are all the other lines in the skeletons.

File tailoring statements and print utility records may contain the names of:

ISPF variables and **table elements**. They are prefixed by an ampersand [&].

File tailoring statements

These are the lines that guide the ISPF file tailoring services through the print utility skeletons, according to the type of print request.

)SEL *condition* paired with)ENDSEL	Select lines to be processed if the specified condition is true. Pairs of)SEL -)ENDSEL statements may be nested.
)SET	Set a variable to a specified value.
)DOT *tablename* paired with)ENDDOT	Include table elements to be processed from table *tablename*.
)CM *comments*	Comment lines.

Note: For details refer to the ISPF Dialog Management guides.

Print utility records

The print utility records in the print utility skeletons are basically structured like the logical lines in the print input file. However, print utility records contain the names of variables and table elements, whereas the logical lines already contain their actual contents.

Note: For the structure of the print input file and the relationship of the print utility records to the lines of the printed output, refer to “The print input file” on page 105.

The following is a description of the print utility records used to create the logical lines of the print input file:

General control record: The general control record specifies the print parameters. This record is mandatory. It must be the first record in the print input file.

```
G ppp wid dep d datasetname membername
```

G	Identifies the record as a general control record.
ppp	Placeholder for the page number. If the string specified here is found in the heading line, it is replaced with the actual page number.
wid	Specifies the page width. The value must be be at least 20. For the system printer and a DBCS printer, the first byte is used as the ASA control character. For a GDDM printer, the last byte is used as the stopping attribute. When output is written to a file, this file has variable record format and record length as specified here.
dep	Specifies the physical page depth. The value must be at least 20. For DCF output, the page depth is assumed to be infinite, regardless of any definition here.
d	Specifies the destination: where 1=system printer, 2=GDDM print utility, 3=DCF output, 4=DBCS printer, and 5=DBCS with outlining.
datasetname	Is the name of the output data set. For destination=3 it must be specified; otherwise it must be blank.
membername	Is the member name of the output file. For destination=2 specify the printer ID here. For destination=3 specify member name here. For any other destination, this parameter is ignored.

DCF control records: The DCF control records allow you to specify strings to be inserted under specified conditions. These records, which are optional, must follow the general control records.

```
DCC c
DCL l
DCK oe
DPxnn<..................(nn <= 75)...................>
DAxihcknn<..............(nn <= 71)...................>
```

DCC *c*	Specifies the DCF continuation character *c* (default +), which is inserted at the end of a record if a line is spilled.
DCL *l*	Specifies the DCF new line character *l* that, when found in a replacement string, forces the start of a new line. This is used when inserting control words (such as a control word for changing the font for highlighting) because DCF control words must begin on a new line.
DCK *oe*	Specifies for DCF the replacement characters *oe* (default blank), for odd and even DBCS characters (default blank). If the *oe* parameter is set to blank or if the control card is missing, no translation takes place.
DP*xnn*	Specifies the string to be inserted at the beginning or end of the panel line. *x* B=Insert at the beginning. The default is 1 blank character. E=Insert at the end of the panel line. The default is no insertion. *nn* The length of the subsequent string.

DA*xihcknn* Specifies the string to be inserted at the beginning or end of an attribute zone.

x B=beginning, E=end
i Intensity (N=normal, I=invisible, B=bright)
h Highlight (N=normal, B=blink, R=reverse, U=underscore)
c Color (D=default, B=blue, R=red, P=pink, G=green, T=turquoise, Y=yellow, N=neutral)
k K=DBCS, E=EBCDIC
nn The length of the subsequent string.

Any value for *i*, *h*, *c*, and *k* not explicitly mentioned, could default to any of the mentioned values.

The specified string is inserted at the start or end of an attribute zone with appropriate properties in a panel line.

Heading records: Originally the heading line is set to blank. The heading records allow you to modify the heading line.

Logically, the heading line is divided into segments of 30 bytes each. Single segments may be modified with heading records, leaving the other segments of the heading line unchanged.

When a heading record is encountered, output of the previous logical page is forced.

More than one heading record may be used to model the heading line. They take effect in the sequence in which they appear.

A special character combination, defined in the general control record, may be inserted in the heading line for the page number. This character combination is replaced by the appropriate value before printing the page.

```
H   <.....................(76).....................>
Hn  <.......(30).......>
Hnm <.......(30).......><.......(30).......>
H0
H-
```

H Clear the heading line, then insert 76 bytes supplied to heading line.

H*n* Overlay the *n*th section of the heading line with 30 bytes supplied, where *n* is in the range of 1 through 9.

H*nm* Overlay *n*th section of heading line with the first 30 bytes supplied. Overlay the *m*th section of the heading line with the second 30 bytes supplied, where *n* and *m* are in the range of 1 through 9.

H0 Do not change heading line. The contents of the heading line are not affected. Output pages continue to be numbered. The only effect is to signal the start of a new page.

H- Suppress output of the heading line. The contents of the heading line are not affected. Output pages continue to be numbered. Output of heading lines is resumed after the next heading line record.

Subheading records and footing records: More than one subheading line or footing line may be specified. If the logical page is split into more than one physical page, the subheading portion and the footing portion of the page are repeated on subsequent pages.

The subheading lines and footing lines are cleared when a heading record is encountered in the input.

```
Snn <.................(nn <= 76).................>
Snn+<.................(nn <= 76).................>
Fnn <.................(nn <= 76).................>
Fnn+<.................(nn <= 76).................>
```

S*nn*	Begin a new subheading line, *nn* bytes long.
S*nn*+	Add *nn* bytes to the previous subheading line.
F*nn*	Begin a new footing line, *nn* bytes long.
F*nn*+	Add *nn* bytes to the previous footing line.

Text line records: Text line records are used to specify plain text lines. Plain text lines contain the text to be printed unchanged. Plain text lines that exceed the page width are truncated.

```
 <.....................(79).....................>
*<.....................(79).....................>
+<.....................(79).....................>
Tnn <.................(nn <= 76).................>
Tnn+<.................(nn <= 76).................>
```

blank	Begin a new text line with a length of 79 bytes.
*	Begin a new text line with a length of 79 bytes. This is useful for creating blank lines on the listing. Blank lines would otherwise be suppressed by ISPF file tailoring.
+	Add 79 bytes to the previous text line.
T*nn*	Begin a new text line with a length of *nn* bytes.
T*nn*+	Add *nn* bytes to the previous text line.

Attribute records: Attribute records may follow any kind of record, except panel records. They specify byte-for-byte the attributes for the corresponding positions of the previous record.

```
Ann <.................(nn <= 76).................>
Ann+<.................(nn <= 76).................>
```

A*nn*	Begin a new attribute line, *nn* bytes long.
A*nn*+	Add *nn* bytes to the previous attribute line.
	One attribute byte is delivered for each text byte. The bits of an attribute byte have the following meanings (1=most significant bit, 8=least significant bit):

```
1      Unused
2,3    Intensity:    0=dark, 1=normal, 2=bright
4,5    Highlighting: 0=normal, 1=blink, 2=reverse, 3=underscore
6,7,8  Color:        0=default, 1=blue, 2=red,
                     3=pink, 4=green, 5=turquoise,6=yellow, 7=white.
```

Panel records: Panel records describe a panel layout. Panel lines that exceed the page width are split vertically, except for DCF output. For DCF output, panel lines are enclosed in the panel-line-begin and panel-line-end strings, as specified in the general control records.

Panels also contain attributes, which, depending on the abilities of the printer, may be reflected in the output.

```
PB wid dep
PL<.....................(78).....................>
PA<.....................(78).....................>
PR<.....................(78).....................>
```

PB *wid dep*
: Start of panel information, where *wid* and *dep* specify the dimension of the panel.

PL
: Panel line text for the complete panel. Each record is 78 bytes long. These records, which are required for panels, must follow the start of panel information record. They must describe the complete panel.

PA
: Attribute information for the complete panel. Each record is 78 bytes long. These records, which are optional, must follow the panel line text records. One attribute byte is delivered for each text byte. The bits of an attribute byte have the following meanings (1=most significant bit, 8=least significant bit):

```
1      Unused
2,3    Intensity:    0=dark, 1=normal, 2=bright
4,5    Highlighting: 0=normal, 1=blink, 2=reverse, 3=underscore
6,7,8  Color:        0=default, 1=blue, 2=red, 3=pink, 4=green,
                     5=turquoise, 6=yellow, 7=white.
```

PR
: Field ruling information for the complete panel. Each record is 78 bytes long. These records, which are optional, must follow the panel line text records or the panel attribute records. One field ruling byte is delivered for each text byte. The bits of a field ruling byte have the following meanings (1=most significant bit, 8=least significant bit):

```
1      0=EBCDIC, 1=DBCS, (also set by program for SO/SI)
2      0=First byte, 1=second byte, of DBCS character
3      Field ruling begin (set by program)
4      Field ruling end (set by program)
5      Field ruling left
6      Field ruling over
7      Field ruling right
8      Field ruling under.
```

ISPF variables and table elements

ISPF variables and table elements are replaced by their assigned values during the first pass of the print utility.

Figure 58 lists the names of the ISPF variables and tables, their meanings, and the types of output for which they are used. In the skeletons, the names of variables and table elements are prefixed by an ampersand (&).

Figure 58 (Page 1 of 2). ISPF dialog variables and tables, and their meanings

For all types of output	
IUV14OSY	Operating system (MVS)
IUV14FMT	Print format **1** system printer **2** GDDM **3** DCF **4** DBCS printer **5** DBCS printer, with outlining
IUV14DSN	Data set name when IUV14FMT=3, else blank
IUV14MEM	Member name when IUV14FMT=3, printer ID when IUV14FMT=2
Variables available except for DGIUS1IG	
IUV14PPN	Name of the object to be printed
IUV14PTY	Object type **G** panel group **P** panel **S** partition set **A** AID table **B** control table
IUV14PLB	Object library
IUV14TSY	Object target system **C7** CICS **M1** MFS **I2** ISPF **G5** GDDM **X2** CSP ***** All
IUV14CMA	Used in some skeletons to select different parts of an output
For output of panels	
IUV14PFN	Format name for the target system MFS or All
IUV14ANM	The name of the scrollable area for the target system ISPF or All
IUV14PDV	Device type
IUV14PPP	Physical page for the target system MFS or All
For output of formats and layouts (DGIUS1P1, DGIUS1G5, DGIUS1S3)	
IUV10PDE	Panel depth

Figure 58 (Page 2 of 2). ISPF dialog variables and tables, and their meanings	
IUV10PWI	Panel width
IUI10PAN	ISPF table containing the panel lines and attribute lines
IUI10PLI	ISPF table element containing the panel lines
IUI10PA1	ISPF table element containing the attributes for lines
IUI10PA2	ISPF table element containing the ruling attributes for lines
For non-tabular output	
Use dialog variables. These dialog variables have the same names as the dialog variables in the associated panels. The names of the associated panels are given in the headers of the skeletons.	
For tabular output	
Use ISPF tables with appropriate table elements. Table IUI10TXT with table elements IUI10T*nn* (*nn*=01, 02, ...) holds the contents of the columns. The maximum length of these table elements is 32 bytes. Skeleton DGIUS1W1 uses the prototype table directly. The names of the columns are given in the header of the skeleton.	

Example of adapting a skeleton

Assume that you want to display the output for the system printer on a terminal that can display only 80 characters on each line. You therefore need to split lines and use continuation lines.

DGIU1P5 is used to print the panel fields. Figure 60 on page 104 through Figure 64 on page 105 show what has to be changed in this skeleton for this example.)CM (comment) statements are included to document the changes.

Unchanged skeleton: Figure 59 on page 101 shows part of the skeleton DGIUS1P5. The parts where the changes have to be made for this example are highlighted.

```
)CM  PRINT UTILITY - Panel fields
)CM
)CM  *=============================================================*
)CM  *                                                             *
)CM  *         This skeleton does NOT require NLS translation      *
)CM  *                                                             *
)CM  *=============================================================*
)CM
)CM
)CM  *-column name-------------length---table columns---target systems-*
)CM  * name                      33         1 2              *CMIGX    *
)CM  * ref name                   2         3                *CMIGX    *
)CM  * line                      3         4                *CMIGX    *
)CM  * column                    3         5                *CMIGX    *
)CM  * depth                     3         6                *CMIGX    *
)CM  * width                     4         7                *CMIGX    *
)CM  * occurrences               3         8                *CMIGX    *
)CM  * array type                8         9                *CMIGX    *
)CM  * field type                8         10               *CMIGX    *
)CM  * scan?                     1         11               *  I      *
)CM  * control table             8         12               * M       *
)CM  * related field            65         13 14 15         *CMIGX    *
)CM  * help name                 8         16               *  I      *
)CM  * field mark                1         17               *CMIGX    *
)CM  *-------------------------------------------------------------*
)CM
)CM  *-------------------------------------------------------------*
)CM  * table except for dcf output                               *
)CM  *-------------------------------------------------------------*
)CM  *-----------------------------------------------------------*
)CM  * table header, underlining and blank line                  *
)CM  *-----------------------------------------------------------*
)CM set the area name
)SET IUV1AREA = &Z
)SET IUV1HEAD = &IKKXH1P5
)SEL &IUV14CMA = PA
)SET IUV1AREA = (&IUV14ANM)
)SET IUV1HEAD = &IKKXH1R5
)ENDSEL

)SEL &IUV14FMT ¬= 3
H2  &IUV1HEAD
)SEL &IUV14TSY = I2 | &IUV14TSY = X2
H3  &IUV14PPN &IUV14PTY &IUV14PLB
S00
S50 &IUV14ANM
S00
)ENDSEL
)SEL &IUV14TSY = C7 | &IUV14TSY = G5
H3  &IUV14PPN &IUV14PTY &IUV14PLB / &IUV14PDV
S00
S00
S00
)ENDSEL
```

Figure 59 (Part 1 of 3). Example of a print utility skeleton

```
)SEL &IUV14TSY = M1 | &IUV14TSY = *
H3  &IUV14PPN &IUV14PTY &IUV14PLB
S00
S50 &IUV14PFN &IUV14PDV &IUV14PPP &IUV1AREA
S00
)ENDSEL
S33 &IKKCPNAM
S07+ &IKKCPREF
S07+ &IKKCPLIN
S07+ &IKKCPCOL
S07+ &IKKCPDEP
S07+ &IKKCPWID
S07+ &IKKCPOCC
S09+ &IKKCPARR
S09+ &IKKCPFTY
)SEL &IUV14TSY = I2 | &IUV14TSY = *
S07+ &IKKCPSCA
S09+ &IKKCPHLP
)ENDSEL
)SEL &IUV14TSY = M1 | &IUV14TSY = *
S09+ &IKKCPOPC
)ENDSEL
S61 ================================ ====== ====== ====== ======
S32+ ====== ====== ======== ========
)SEL &IUV14TSY = I2 | &IUV14TSY = *
S07+ ======
S09+ ========
)ENDSEL
)SEL &IUV14TSY = M1 | &IUV14TSY = *
S09+ ========
)ENDSEL
S00
)CM  *-------------------------------------------------------------*
)CM  * table rows                                                  *
)CM  *-------------------------------------------------------------*
)SET IUV1ENT = N
)DOT IUI10TXT
)SEL &IUI10T13 ¬= &Z
)SET IUV1ENT = Y
)ENDSEL
T32 &IUI10T01
T01+&IUI10T02
T07+ &IUI10T03
T07+ &IUI10T04
T07+ &IUI10T05
T07+ &IUI10T06
T07+ &IUI10T07
T07+ &IUI10T08
T09+ &IUI10T09
T09+ &IUI10T10
)SEL &IUV14TSY = I2 | &IUV14TSY = *
T07+ &IUI10T11
T09+ &IUI10T16
)ENDSEL
```

Figure 59 (Part 2 of 3). Example of a print utility skeleton

```
)SEL &IUV14TSY = M1 | &IUV14TSY = *
T09+ &IUI10T12
)ENDSEL
)ENDDOT
)CM  *-------------------------------------------------------------------*
)CM  * table except for dcf output              - part 2              *
)CM  *-------------------------------------------------------------------*
)CM  *-----------------------------------------------------------------*
)CM  * table header, underlining and blank line  - part 2           *
)CM  *-----------------------------------------------------------------*
)SEL &IUV1ENT = Y
S00
S00
S00
S33 &IKKCPNAM
S66+ &IKKCPRFD
S33 ================================
S66+ ================================================================
S00
)CM  *-----------------------------------------------------------------*
)CM  * table rows                                - part 2           *
)CM  *-----------------------------------------------------------------*
)DOT IUI10TXT
)SEL &IUI10T13 ¬= &Z
T32 &IUI10T01
T01+&IUI10T02
T33+ &IUI10T13
T32+&IUI10T14
T01+&IUI10T15
)ENDSEL
)ENDDOT
)ENDSEL

)ENDSEL
)CM  *-------------------------------------------------------------------*
)CM  * table for dcf output                                             *
...
...
...
```

Figure 59 (Part 3 of 3). Example of a print utility skeleton

Changes to skeleton DGIUS1P5: Figure 60 on page 104 to Figure 64 on page 105 show the changed parts of the skeleton DGIUS1P5, in the same sequence as the original, highlighted parts in Figure 59 on page 101.

1. Shorten the heading. The third section of the heading line (H3) is positions 61 through 90. Because it can not be more than 20 characters long, move the device type (IUV14PDV) to the subheading line. Figure 60 on page 104 shows what the code should now look like.

```
H3  &IUV14PPN &IUV14PTY &IUV14PLB
S00
S00 &IUV14PDV
S00
```

Figure 60. Shortened heading

2. Split the table heading into two lines. In the first line, keep all fields up to IKKCPOCC (occurrence number). Move the rest to the second line. Insert 33 blanks at the beginning of the second line, to align with IKKCPREF (reference name). Figure 61 shows what the code should now look like.

```
S07+ &IKKCPOCC
)CM second subheader line
S33
S09+ &IKKCPARR
```

Figure 61. Split table heading

3. Delete the gaps from the separator line. Because the gaps in the two heading lines are no longer aligned, the gaps in the separator line are unnecessary. Figure 62 shows what the code should now look like.

```
)ENDSEL
S40 ========================================
S36+===================================
S00
```

Figure 62. Gaps deleted from the separator line

4. Split the table rows into two lines, in the same way that you split the table heading. Figure 63 shows what the code should now look like.

```
T07+ &IUI10T08
)CM second table line
T33
T09+ &IUI10T09
```

Figure 63. Split table rows

5. Split the lines in the second part of the table. Indent the related field by 14 blanks, to make the lines no longer than 80 characters. Figure 64 on page 105 shows what the code should now look like.

```
S33 &IKKCPNAM
)CM second subheader line
S14
S66+ &IKKCPRFD
S40 ========================================
S40+========================================
S00
)CM  *---------------------------------------------------------------*
)CM  * table rows                                 - part 2           *
)CM  *---------------------------------------------------------------*
)DOT IUI10TXT
)SEL &IUI10T13 ¬= &Z
T32 &IUI10T01
T01+&IUI10T02
)CM second table line
T14
T33+ &IUI10T13
```

Figure 64. Split lines in the second part of the table

The print input file

During the first pass of the SDF II print utility, the ISPF file tailoring services create the print input file based on the file tailoring statements of the appropriate print utility skeletons. File tailoring services replace the variables and table elements with their assigned values.

The records of the print input file describe the lines of the printed output. More than one record of the print input file can be used to describe one line of the printed output. For example, the information of three heading records can be concatenated to form one heading line of the printed output.

The print input file consists of fixed-length records, 80 bytes long.

Figure 65 on page 106 shows the records of the print input file.

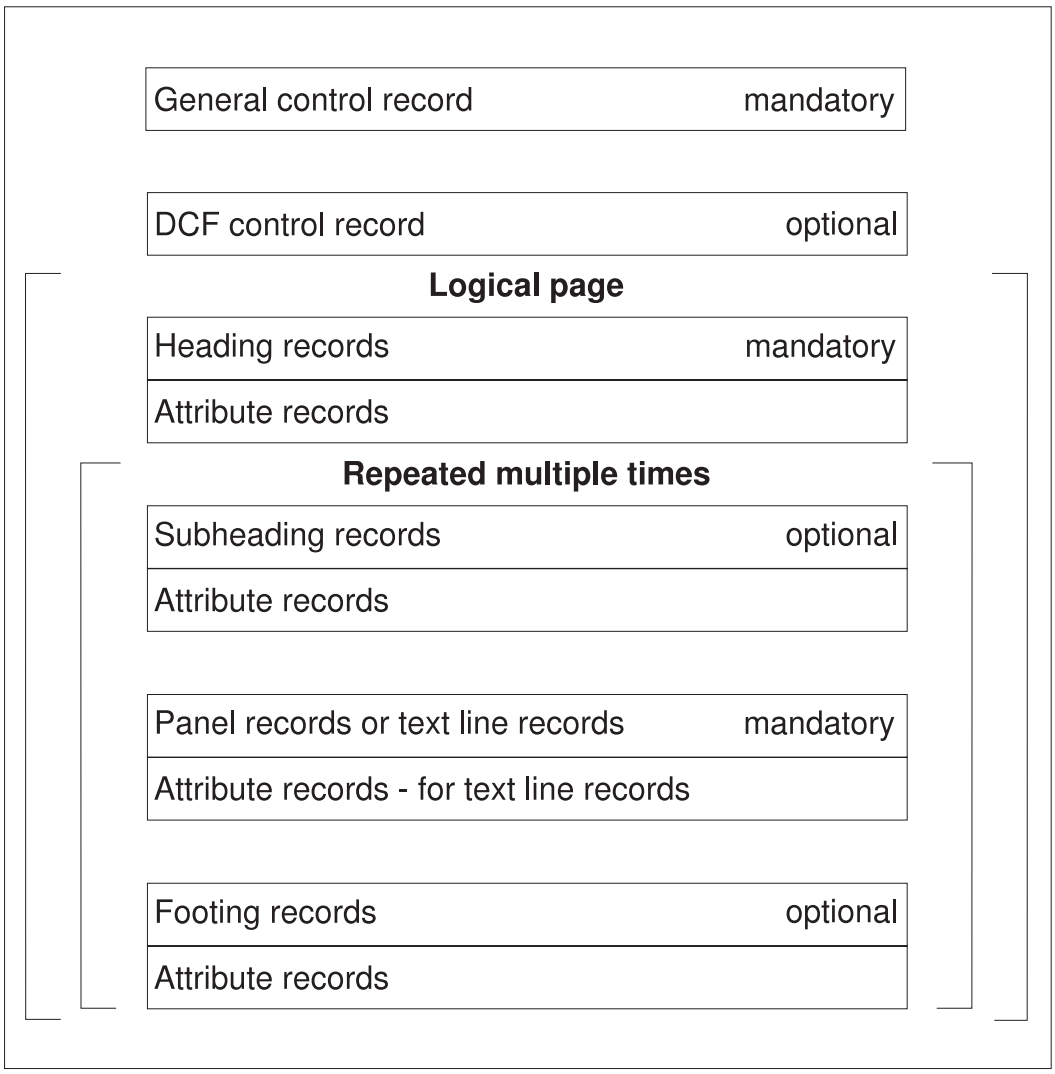


Figure 65. The structure of the print input file

The width of a logical page may be specified in the general control record at the beginning of the print input file. Records exceeding the defined width are truncated. The logical page depth is not restricted.

For panel output, records that exceed the page width are continued if the output is for DCF. Otherwise, panels are split into multiple pages.

A logical page of the print input file has the following records:

- The **heading record** is the first record of a logical page. It may contain the placeholder for the page number. The heading record is required. It triggers the beginning of a new logical page.
- **Subheading records**, which are optional, remain effective until the next block of subheading records. More than one subheading record may exist.
- **Panel records** or **text line records**

  Text line records contain the text to be printed unchanged. Text line records that exceed the page width are truncated.

Panel records describe formats. Panel line records that exceed the page width are continued if the output is for DCF. Otherwise, formats exceeding the page width are split vertically.

If neither panel records nor text line records exist, the output of the page is suppressed.

- **Footing records**, which are optional, take effect from the end of the previous block of subheading records. More than one footing record may exist.
- **Attribute records** may follow any record, except panel records. They immediately follow the record they describe.

Subheading records, text line records, and footing records may be intermixed freely. Each record of a print input file may contain DBCS strings enclosed in SO and SI characters. They must not wrap lines except for panels, where wrapping is allowed.

In the second pass of the SDF II print utility, the lines of the printed output are built according to the specifications in the print input file.

End of General-use programming interface

Changing the uppercase/lowercase translation tables

General-use programming interface

In some SDF II dialogs, you can use these line commands:

u For translation to uppercase
l For translation to lowercase.

In the uppercase translation table, you find for each hexadecimal code the corresponding hexadecimal code that is used when you issue the **u** line command.

Figure 66 on page 108 and Figure 67 on page 108 show the assembler code of the tables that are used for this translation.

The uppercase translation table, DGIYTUC and the lowercase translation table, DGIYTLC are in SDF2.V1R4M0.SDGISAM.

```
*********************************************************************
* DGIYTUC - TRANSLATE TO UPPER CASE TRANSLATE TABLE
*********************************************************************
DGIYTUC  CSECT ,  0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'000102030405060708090A0B0C0D0E0F'    0.
         DC    X'101112131415161718191A1B1C1D1E1F'    1.
         DC    X'202122232425262728292A2B2C2D2E2F'    2.
         DC    X'303132333435363738393A3B3C3D3E3F'    3.
         DC    X'404142434445464748494A4B4C4D4E4F'    4.
         DC    X'505152535455565758595A5B5C5D5E5F'    5.
         DC    X'606162636465666768696A6B6C6D6E6F'    6.
         DC    X'707172737475767778797A7B7C7D7E7F'    7.
         DC    X'80C1C2C3C4C5C6C7C8C98A8B8C8D8E8F'    8.
         DC    X'90D1D2D3D4D5D6D7D8D99A9B9C9D9E9F'    9.
         DC    X'A0A1E2E3E4E5E6E7E8E9AAABACADAEAF'    A.
         DC    X'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'    B.
         DC    X'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'    C.
         DC    X'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'    D.
         DC    X'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'    E.
         DC    X'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'    F.
         END   DGIYTUC
```

Figure 66. Table for translation to uppercase

For example, if you want to find the uppercase value for hexadecimal code 81, look along the line marked 8 on the right of the table. Then look down the column marked 1 on the top of the table. Where line 8 and row 1 intersect, you find the value `C1`.

You read the lowercase translation table in the same way.

```
*********************************************************************
* DGIYTLC - TRANSLATE TO LOWER CASE TRANSLATE TABLE
*********************************************************************
DGIYTLC  CSECT ,  0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'000102030405060708090A0B0C0D0E0F'    0.
         DC    X'101112131415161718191A1B1C1D1E1F'    1.
         DC    X'202122232425262728292A2B2C2D2E2F'    2.
         DC    X'303132333435363738393A3B3C3D3E3F'    3.
         DC    X'404142434445464748494A4B4C4D4E4F'    4.
         DC    X'505152535455565758595A5B5C5D5E5F'    5.
         DC    X'606162636465666768696A6B6C6D6E6F'    6.
         DC    X'707172737475767778797A7B7C7D7E7F'    7.
         DC    X'808182838485868788898A8B8C8D8E8F'    8.
         DC    X'909192939495969798999A9B9C9D9E9F'    9.
         DC    X'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'    A.
         DC    X'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'    B.
         DC    X'C081828384858687888 9CACBCCCDCECF'    C.
         DC    X'D091929394959697 9899DADBDCDDDEDF'    D.
         DC    X'E0E1A2A3A4A5A6A7A8A9EAEBECEDEEEF'    E.
         DC    X'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'    F.
         END   DGIYTLC
```

Figure 67. Table for translation to lowercase

If you want to change something in the translation tables:

1. Retrieve the members DGIYTUC and DGIYTLC from the example library SDF2.V1R4M0.SDGISAM.
2. Adapt the contents of the tables, as appropriate.
3. Assemble the tables and link them into the partitioned data set SDF2.V1R4M0.ADGIMOD1.
4. Use the member DGITRANS from the library SDF2.V1R4M0.SDGIJCL to link SDF II, in order to make use of the adapted translation tables.

Notes:

1. Always keep copies of the original tables in case you want to use the original SDF II defaults.
2. Create a backup copy of the customized tables because any program maintenance may change them.

End of General-use programming interface

Changing ISPF keylist

In SDF II, PF key assignments are made by an ISPF keylist table in library SDF2.V1R4M0.SDGITBxx (where xx is the language code), which contains all keylists used in SDF II panel definitions. The source for this keylist definitions is in library SDF2.V1R4M0.SDGISAM(DGIKEYS). Modifications to any keylist must be made in source form and the complete definition must be compiled with the ISPF Dialog Tag Language Compiler (DTLC) to get a new keylist table.

Follow these steps to make changes to keylist table:

- Copy SDF2.V1R4M0.SDGISAM(DGIKEYS) to a private library.
- Apply changes.
- Compile source keylist definition with DTLC. Route output to a private table library.
- Concatenate private table library to ISPTLIB as described in “General definitions” on page 6.

Chapter 6. Importing objects into SDF II

This chapter describes how to use the import utility to import objects from various sources into SDF II libraries.

The import utility allows users to import the following into an SDF II supported library:

- Maps, map sets, and partition sets defined with CICS/BMS macros
- Format sets defined with IMS/MFS utility control statements
- Panels defined in ISPF panel syntax
- Maps, map groups, and AID tables defined by and exported from GDDM-IMD
- Maps and map groups defined by and exported from CSP/AD
- Maps, map sets, and partition sets defined and dumped with SDF/CICS.

Note: "Development environments" on page xv lists the IBM licensed programs from which you can import object definitions into SDF II.

Select the import utility, which is option **9.2** on the Select an SDF II Function panel or option **2** on the Select a Utility panel. The Specify Import Utility Parameters panel is then displayed. Define, on this panel, the input that is to be imported and the library in which the imported object is to be stored.

Importing objects from CICS/BMS

Use the import utility to create SDF II panels, panel groups, and partition set objects from CICS/BMS macro statements. The import utility imports one file of map set macros or partition set macros. The utility assumes that the input consists of correct macros.

The following checks are made:

- The first macro statement is checked to make sure that it is either a `DFHMSD` macro for map sets or a `DFHPSD` macro for partition set definitions.
- Macro parameters are checked syntactically and, if any required parameters are found to be missing, a message is issued.

Comments and remarks on the macro statements are ignored, unless they are data structure name definition records. These records can be added as comment lines to the CICS/BMS macros to be imported. These records are used by the import utility to build data structure names. For the format of the data structure records refer to "Importing data structure names" on page 115.

If an object with the same name already exists, a new instance is created if the current device is different from the devices for which the existing object is defined. An SDF II object is created only if the macro statements are correct.

Panel and panel group import

SDF II panels and panel groups are created from CICS/BMS DFHMSD, DFHMDI, and DFHMDF macro statements.

The input is a file of CICS/BMS macro statements. These statements define a map set that starts with a DFHMSD statement. The DFHMSD statement contains one or more maps that start with a DFHMDI statement. The last statement must be DFHMSD TYPE=FINAL.

One panel group is created for each map set, one panel is created for each map.

Partition set import

An SDF II partition set is created from CICS/BMS DFHPSD and DFHPDI macro statements.

For any missing optional parameters, the SDF II partition set editor defaults are used.

Checks are made that:

- The viewports are correctly positioned in the usable area, and that the size of the viewports is correct.
- Partitions do not overlap.
- For the IBM 8775 display terminal, the partition buffer, depth buffer, and scroll storage limits are correct.

Importing objects from IMS/MFS

Use the import utility to create SDF II panels, AID tables, partition sets, and control tables from IMS/MFS utility control statements.

Data structure name definition records can be added as comment lines to the IMS/MFS utility control statements to be imported; they are used by the import utility to build data structure names. For the format of the data structure records refer to “Importing data structure names” on page 115.

Before you import any IMS/MFS object, follow these steps:

1. Verify that each data set that contains a format definition also contains all the message definitions that reference this format on the SOR parameter. Format definitions begin with an FMT statement. Message definitions begin with an MSG statement. The SOR parameter defines the format to which the message belongs. This step is necessary because import can be performed only on complete format sets.
2. Verify that each data set that contains a partition descriptor also contains at least one DEV statement for each device type with which the partition descriptor will be used. The DEV statements should specify the partition descriptor on the PDB parameters. Partition descriptors begin with a PDB statement. This step is necessary because the import utility requires details of the devices on which the partition descriptor will be used.

You should also make sure that the IMS/MFS input obeys the restrictions listed in “Restrictions” on page 114.

The import utility assumes that the IMS/MFS input to be imported is valid. You can verify this by checking whether the IMS/MFS language utility gives a return code of 4 or less when it interprets the input. A return code 4 is probably caused by an error that is not connected with the source statements. These errors produce such messages as DFS1039 or DFS1014.

The import utility checks for errors, but not as comprehensively as does the MFS language utility.

If you attempt to import invalid IMS/MFS input, unpredictable results will occur. The import utility may accept incorrect input, but not issue an error message, which produces invalid SDF II objects.

The import utility checks the following error conditions:

- Violation of the restrictions listed under “Restrictions” on page 114
- Most syntax errors
- Conditions that would cause the import utility or another SDF II component to terminate abnormally.

Panel and AID table import

The import utility creates one SDF II panel for each format set in the IMS/MFS input. The panel name is the label on the FMT statement.

The import utility also creates one AID table for each DEV statement that specifies the program function (PF) key parameter. The AID table name is the format set name with a 2-digit suffix added. This suffix is the number of the DEV in the format definition. For example, the AID table for the second DEV in the format FMTX will have the name FMTX02.

The IMS/MFS input must contain the format definition and all message definitions in the format set. This is necessary because import can be performed only on complete format sets.

Partition set import

The import utility creates one SDF II partition set for each partition descriptor in the IMS/MFS input. The partition set name will be the label on the PDB statement.

The IMS/MFS input must also contain at least one DEV statement for each device type with which the partition descriptor is used. The DEV statements must specify the partition descriptor on the PDB parameters. This is necessary because the import utility requires details of the devices with which the partition set will be used.

When you use the partition set editor, the partitions are renumbered, starting with partition identification 0.

Control table import

The import utility creates one SDF II control table for each control table in the IMS/MFS input. The name of the control table is the label of the TABLE statement.

Use of the SDF II device table

The import utility uses the SDF II device table to retrieve information about devices specified by DEV statements. It searches the table for an entry whose MFS device type and, in some cases, page width match those that are specified on the DEV statement.

The search is performed as follows:

1. If the DEV TYPE parameter is omitted, it is assumed to be 3270,2.
2. For device types that match, leading zeros in the numeric part of suffixes are ignored. For example, 3270-A02 is considered the same as 3270-A2. Also, the old and new forms of the finance work station device types are considered equivalent. For example, 3600 is considered the same as FIN.
3. If the IBM device type is a 3270P, FIFP, SCS1, or SCS2, for a match to be found, the width in the device table entry must be the same as or greater than the width specified on the WIDTH or FEAT parameter. If these parameters are omitted, or if the device type is not one of those listed above, the width is determined from the device table.

If a matching entry cannot be found, the DEV statement is ignored.

Restrictions

The following restrictions apply to the import of IMS/MFS input. The restrictions that are most likely to be violated are listed first:

- Only complete format sets can be imported. This means that a format definition should be imported together with all of the message definitions that specify the format on the SOR parameter. If a format set is imported without a format definition or without any message definitions, the import utility displays an error message and ignores the format set.
- A partition descriptor should be imported with at least one DEV statement for each device type with which the partition descriptor will be used. The DEV statements must define the partition descriptor on the PDB parameters. If a partition descriptor is imported without any DEV statements referencing it, the import utility displays an error message and ignores the partition descriptor.
- A partition descriptor must not be referenced by more than three DEV statements with different device types. Otherwise, the import utility displays an error message and ignores the extra references.
- Identifiers must contain only uppercase alphabetic, numeric, and national characters. They may start with an alphabetic character (including special national characters). IMS/MFS identifiers may contain ampersands and any characters defined on preceding ALPHA statements. If an identifier contains one of these characters, the import utility displays a warning message and translates the invalid characters into valid characters.
- There should be at least six graphic characters or digits that can be used as field marks in the panel. A character can be used as a field mark if it does not appear in any DFLD literal. This does not include DFLDs in DEVs with TYPE=DPM or MODE=STREAM. If there are not enough characters available, the import utility displays an error message and does not create the panel.
- A format definition cannot contain more than 99 DEV statements. If it does, the import utility displays an error message and ignores the extra DEV statements.

- The import utility always interprets the characters X'0E' and X'0F' in a literal as shift characters for mixed DBCS. (IMS/MFS allows them to be defined as alphabetic characters and then be used in literals as EBCDIC characters.)
- Only valid IMS/MFS input can be imported.

Importing data structure names

Use the import utility to create data structure names from records that you add as comment lines to CICS/BMS macros or IMS/MFS utility control statements to be imported.

During import, SDF II will use the information in these records to assign data structure names to imported fields. The imported data structure names will later be used for the generation of the panel's data structures.

You can define data structure names for:

- Fields, arrays, CICS/BMS groups, and subfields for CICS/BMS
- MFS segments and fields for IMS/MFS.

For CICS/BMS maps and map groups it is also possible to create arrays and repeat formats using data structure name records.

These types of records are necessary to create a data structure name definition:

*DGI*DSECT	First record.
DGI	One or more element records.
*DGI*REP*	Repeat element record; optional. Used to create a repeat format in the imported SDF II panel (CICS/BMS only).
*DGI*END	Last record.

The structure of one data structure name definition is as follows:

```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*DGI*DSECT mmmmmmmm lpagenam                                          00000010
*DGI*lv length occ dsect_name                                         00000020
*DGI*...                                                              000000..
*DGI*REP*occ row col dep wid dsect-name                               000000..
*DGI*...                                                              000000..
*DGI*END                                                              00000090
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
```

The element records that define one data structure must not be interrupted by any lines, except for comment lines or blank lines, which are ignored by SDF II.

Figure 68 shows the structure of the *DGI*DSECT record:

Figure 68. Begin record

Columns	Description
1-10	*DGI*DSECT
11	blank
12-19	Map name for CICS/BMS MSG name for IMS/MFS
20	blank
21-28	LPAGE name (IMS/MFS only, optional)
29-72	blank
73-80	Sequence number

Figure 69 shows the structure of the **element record**:

Figure 69. Element record

Columns	Description
1-5	*DGI*
6-7	A two-digit level number. Possible values are: 1 MFS segment 2 MFS field BMS field, BMS group, BMS array 3 BMS subfield in a group BMS field, BMS group, BMS array within a repeat format 4 BMS subfield of a group within a repeat format
8	blank
9-14	Length of field (1 ... 256 for BMS, 1 ... 8000 for MFS). Must be blank for MFS segments and for BMS groups.
15	blank
16-18	Occurrence number (1 ... 999 for BMS only) or blank
19	blank
20-71	Name of the data structure element or blank (SDF II currently supports names up to 31 characters long)
72	blank
73-80	Sequence number

Figure 70 shows the structure of the **repeat record** (available for CICS/BMS only):

Figure 70. Repeat element record

Columns	Description
1-9	*DGI*REP*
10-12	Occurrence number of repeat format (1...999)
13	blank
14-16	Row number in which the repeat format should start (1...999)
17	blank
18-20	Column number in which the repeat format should start (1...999)
21	blank
22-24	Depth of the repeat format (1...999)
25	blank
26-28	Width of the repeat format (1...999)
29	blank
30-71	Name of the repeat format or blank. (SDF II currently supports names up to 31 characters long)
72	blank
73-80	Sequence number

Rules for CICS/BMS

When you import from CICS/BMS, observe the following:

- The fields of one or more maps can be named. One data structure name definition is required per map.
- Each definition must be within the same file as the CICS/BMS macros defining the map.
- A data structure name definition for a map can be specified at any place of the file, if a map name is specified in the begin record.
- If no map name is specified in the begin record, it will be associated with the map defined by the previous DFHMDI macro. Such a definition cannot be placed before the first DFHMDI macro.
- One element record must be specified for each variable field, CICS/BMS group, or subfield.

  The sequence of element records must be the same as the sequence of the corresponding DFHMDF macros of the variable fields, groups, or subfields.

  The element record for a CICS/BMS group must be specified immediately before the element record that describes the first subfield of this group (a CICS/BMS group has no associated DFHMDF macro).
- The level must be:

  1 MFS segment
  2 MFS field
  BMS field, BMS group, BMS array
  3 BMS subfield in a group
  BMS field, BMS group, BMS array within a repeat format
  4 BMS subfield of a group within a repeat format

- The length:
  - Must not be specified for CICS/BMS groups.
  - Is optional for fields and subfields. If specified, it must match the value specified in the DFHMDF macro.
- The occurrence number for arrays:
  - Must be specified and match the value *n* in OCCURS=*n* if specified in the DFHMDF macro.
  - Can be specified to combine fields or BMS groups into an array. The fields that will be combined into an array,
    - Must be in sequence, without any DFHMDF macros, not belonging to the array, in between
    - Must have the same subfield structure, that is, if the first field is a BMS group, each field to be added to the array must be a BMS group with the same subfield structure as well
    - Must have the same length and no occurrence number
    - Must be identically defined in justification, case, input/output-pictures, and decimal positions.
  - Can be specified to combine subfields of a group into an array. The subfields that will be combined into an array,
    - Must all be part of the same BMS group
    - Must be in sequence, without any DFHMDF macros, not belonging to the array, in between
    - Must have the same length.
- To create a repeat format:
  - Add a repeat element record before the first record that is to be part of the repeat format. The repeat element record must specify the position (row, column), size (depth, width), and repetition factor (occurrence number) of the repeat format. The repeat format must not wrap out of the panel format.
  - All fields that are part of the repeat format have a level that is one higher than for the same construct outside the repeat format:

Construct	Level within	Level outside
Fields, BMS groups, arrays	3	2
Subfields	4	3

  - The structure of the repeat format must be identical in the base part and each repeated part, that is,
    - The number of fields must be the same
    - The sequence of fields must be the same
    - If a field is a BMS group in one part, it must be a group with identical subfield structure in all other parts as well
    - If a repeat format part contains an array, the same array must exist in all other parts as well.
  - Repeated fields must have the same
    - Size
    - Occurrence number
    - Field format (EBCDIC, DBCS or MIXED)

    - Application attributes (justification, case)
    - Input/output pictures, decimal positions.
  - A repeat format cannot contain another repeat format
  - A field that is part of the repeat format must be fully contained in a repeat format part. This includes also the attribute byte in front of the field.
  - Constant field text may vary in the repeated parts.

Rules for IMS/MFS

When you import from IMS/MFS, observe the following:

- The fields of one or more MSG or LPAGE statement can be named. For each MSG/LPAGE statement combination, a separate data structure definition is required.
- Each data structure definition must be in the same file as the IMS/MFS utility control statements that define the format set.
- All data structure definitions must be specified after the END statement of the format set.
- If a MSG statement has no or only one LPAGE statement, the LPAGE statement name can be omitted in the begin record. If the MSG contains more than one LPAGE, an LPAGE name is required in the begin record.
- One element record must be specified for each segment (SEG) and message field (MFLD), except for literals and system literals in output messages.

  The sequence of the element records must be the same as the sequence of the associated SEG and MFLD statements in the corresponding LPAGE or MSG statements.

  Note: For an MFLD statement with a length specification of type (*pp*,*nn*), SDF II generates a minor structure and a field declaration. The specified data structure name applies to the name of the minor structure.
- The level must be:

  1 For segments
  2 For fields.
- The length:
  - Cannot be specified for segments. (If specified, it is ignored.)
  - Is optional for fields. If specified, it must match the value specified in the MFLD statement.
- The occurrence number must not be specified for IMS/MFS.

Importing objects from ISPF

Use the import utility to create an SDF II panel from an ISPF panel. The ISPF panel must be syntactically correct. If the ISPF panel has not been used previously in an ISPF application, it should be checked in PDF test mode.

The import utility builds a mark table that contains at least the definition of the background attributes, a constant field mark, and a variable field mark.

The import utility handles the information in the various sections as follows:

- Information from the NUMBER parameter of the)CCSID section will be stored in the ISPF characteristics.
- If parameters are specified in the)PANEL section, the information they contain will be stored in the ISPF characteristics.
- If an)ATTR section is defined, attribute characters are translated into marks, if possible; otherwise, they are translated into attribute descriptors.

  CUA panel element attribute characters of the type action bar or reference phrase are always translated into marks.
- The information from the)ABC,)ABCINIT, and)ABCPROC sections is stored with the corresponding fields.

  All .ZVARS statements are removed from the)ABCINIT sections; the names contained therein become pull-down field names.
- The)BODY section is translated into the format part of the SDF II panel. ISPF INPUT and OUTPUT fields are translated into variable fields. The length of a variable field is determined by the position of the attribute following the field.

  CUA panel element fields are translated as follows:

  – Fields of the types CEF, LEF, NEF, EE, VOI, LI, and LID are translated into variable fields.
  – Fields of the types AB, ABSL, CH, CT, ET, PT SAC, SI, SUC, WASL, WT, and RP are translated into constant fields.
  – Fields of the types DT, FP, NT, and PIN, are translated either into constant fields or background text.

  TEXT fields are translated either into constant fields or background text. The background text is used when the field has NORMAL intensity and no other attribute has been defined.

  The dynamic and graphic areas are unchanged.
- Any)MODEL section is translated into a repeat format with an occurrence number of 1 or *.
- For the)AREA section, a separate format is created, which will show only the contents of the scrollable area, but no elements of the main format. In the main format only the window in which the scrollable area is to be displayed, will be shown by using the appropriate area mark.
- If an)INIT section is defined, SDF II processes the statements it contains, as follows:

  – Any .ZVARS statement is removed from the)INIT section; the names contained therein become field names.
  – Any .CURSOR and .CSRPOS statements are translated into the cursor attribute.
  – Any .HELP statement is kept as the help panel name in the general panel characteristics.
  – An initialization statement (as in the assignment of a literal to a field variable that has no value) is removed from the)INIT section. The literal is then translated into an initial field value. For example, the code:

    ```
    IF(&FIELD = ' ')
       &FIELD = 'value'
    ```

    is removed from the)INIT section. *value* then becomes the initial value of FIELD.

Any other statements of the)INIT section are added to the list of)INIT section statements.

- All statements of the)REINIT section remain unchanged and are added to the list of)REINIT section statements.
- All statements of the)PROC section remain unchanged and are added to the list of)PROC section statements.
- The information from the)HELP section is stored with the corresponding field.
- All lines of the)END section remain unchanged and are added to the list of)END section lines.

Restrictions

- In the)INIT section, the)REINIT section, and the)PROC section of an ISPF panel, more than one statement can appear on each line.

  Any of the following statements of the)INIT section must be the only statement on a line if it is to be correctly imported:

  .ZVARS
  .CURSOR
  .CSRPOS
  .HELP
  An initialization statement.

- & variables in initial values are not resolved.
- Fields of the type `INPUT` or `OUTPUT` or equivalent CUA-type fields in a scrollable area, must not:
  - Start in column 1 or end in the last column
  - Wrap around lines.

Importing objects from GDDM-IMD

Use the import utility to create panels, panel groups, and AID tables from maps, map groups, and AID tables originally created with GDDM-IMD.

The import utility uses a GDDM-IMD export data set and creates SDF II objects.

Use the GDDM-IMD export utility to create an export data set. You can concatenate several export data sets into one input data set.

For GDDM-IMD, a device type for map and map group import is a unique 2-byte string.

When you import objects from GDDM-IMD, the device table is searched for an SDF II device with a matching GDDM-IMD device code. This code indicates the device type of the created SDF II panel or panel group.

To update the device table use the device table editor, which you can access in the SDF II customization dialog.

If a GDDM-IMD map or map group contains a specification that has more than one device type, the device list in the SDF II device table must contain all the device types of that specification.

SDF II scans the device table for each device type in the specification. It then scans for a device list that contains only these device types. The first device list that fulfills the condition will be the device type of the created SDF II panel or panel group.

Restrictions

If a format contains array elements in a random sequence (that is, if they are ordered neither horizontally nor vertically), the import utility reorders the array elements into a horizontally ordered array and displays a message.

Import of GDDM-IMD maps that have the same name, but belong to different map groups, creates duplicate names in SDF II. These duplicate names are processed as defined in the OPTION field of the Specify Import Utility Parameters panel.

The original names are retained in the description and as the generation names of any imported object.

Tabulator positions defined in GDDM-IMD are removed during import.

Field starter characters, the spacer character, and the graphic area display character may be changed by the import utility.

Arrays of structures that cannot be mapped to a repeat format are ignored.

Importing external source format

Use the import utility to create panels and panel groups from map and map group structures originally created in CSP/AD external source format or in extended external source format (see “Extended external source format”).

The import utility uses the external source format as input and creates SDF II objects.

To create an external source format, use the CSP/AD export utility, VisualGen, or other tools that support this format.

Refer to “Importing CSP/AD export data sets” on page 127 for importing CSP/AD export data sets.

For standard external source format refer to *CSP/AD External Source Format Reference, SH20-6433*.

Extended external source format tags and their attributes are used to represent panel properties that are not supported in CSP/AD, but are supported in CICS/BMS and IMS/MFS.

Extended external source format

For each standard map or map group structure tag, there is a corresponding extended tag. Within an extended tag, all standard and extended attributes are allowed. Within standard tags, no extended attributes are allowed. Within one map or map group structure, standard tags and extended tags cannot be mixed.

If the map or map group structure contains only standard tags, a panel or panel group is created for target system CSP/AD.

If the map or map group structure contains extended tags, a panel or panel group is created for target system ALL.

The following list shows the extended tags and corresponding standard CSP/AD external source format tags:

Extended Tag	Standard Tag
DGIAREA	AREA
DGICATTR	CATTR
DGICFLD, EDGICFLD	CFIELD, ECFIELD
DGIPRSNT	PRESENT
DGIMAP, EDGIMAP	MAP, EMAP
DGIMAPED	MAPEDITS
DGIMAPG, EDGIMAPG	MAPG, EMAPG
DGIMSGS	MESSAGES
DGIVFLD, EDGIVFLD	VFIELD, EVFIELD
DGIVATTR	VATTR
DGIAID, EDGIAID	
DGIPFK	
DGIMFLD, EDGIMFLD	

The extended tags DGIAREA, DGIPRSNT, DGIMAPED, DGIMAPG, and DGIMSGS have the same attributes as the corresponding standard CSP/AD external source format tags.

The extended tags DGICATTR, DGICFLD, DGIMAP, DGIVATTR, and DGIVFLD have extended attributes in addition to the attributes of the corresponding standard CSP/AD external source format tags. These tags are listed here:

DGICATTR and DGIVATTR

The following attributes are supported for target system IMS/MFS only:

SLDI = *nn*
: The line density in lines per inch. The minimum is 1; the maximum is 72.

SLDP = *nn*
: The line density in points per inch. The minimum is 1; the maximum is 72.

DGICFLD

The following attribute is supported for target system IMS/MFS only:

ATTR = **n** | **y**
: **y** specifies that the position in front of this field is reserved for attributes. The external source format field position is the position of the attribute.

  n specifies that the position in front of this field is not reserved for attributes. The field position is the position that would have been used by an attribute. Specification of COLUMN=0 leads in this case to a field with the first field position in column 1. Do not specify **n** unless it is supported by the device.

  The default depends on the device type. It is **n** if the device does not require a separate byte position for the attribute. Otherwise, the default is **y**.

DGIMAP

The following attributes are supported for target systems CICS/BMS and IMS/MFS:

ADJUNCT = **n** | **y**
y specifies that adjuncts for dynamic modification of all attributes will be generated in the data structure for each field in the map.

n specifies that only those adjuncts required by the target system are generated. The default is **n**.

CURSOR = *line column*
Specifies the position where the cursor is to be placed initially.

DEVICES
Within a DGIMAP tag, in addition to the valid CSP/AD devices the following device names are supported:

- 5550
- All 3270-like devices supported by CICS/BMS and IMS/MFS
- 3270P, 3270P1, 3270P2
- SCS1, SCS2.

TITLE = '*text*'
A panel description of up to 32 characters within quotes. Quotes of the same kind as the delimiting quote that are included in the text must be doubled, and each pair of these doubled quotes counts as one character.

The following attribute is supported for target system CICS/BMS only:

FRSET = **n** | **y**
Specifies that the modified data tags of all fields currently in the 3270 buffer are to be reset to a not-modified condition before any map data is written to the buffer.

n specifies that the modified data tags are not to be reset. The default is **n**.

The following attributes are supported for target system IMS/MFS only:

EJECT = *bgnpp*|*endpp*|*bgnmsg*|*endmsg*
Printer form feed value.

FORMS = '*literal*'
A forms literal of up to 16 characters within quotes.

SCA = **n** | **y**
y specifies that the map contains a system control area.

n specifies that there is no system control area. The default is **n**.

SLDI = *nn*
The line density in lines per inch, from 1 to 72.

SLDP = *nn*
The line density in points per inch, from 1 to 72.

TCLIT = '*literal*'
Specifies that the map contains a transaction code. The last character of the literal string should be a blank. If it is not, a blank is added. The maximum length is 256.

DGIVFLD
The following attributes are supported for target system IMS/MFS only:

ATTR = **n** | **y**
y specifies that the position in front of this field is reserved for attributes. The external source format field position is the position of the attribute.

n specifies that the position in front of this field is not reserved for attributes. The field position is the position that would have been used by an attribute. Specification of COLUMN=0 leads in this case to a field with the first field position in column 1. Do not specify **n** unless it is supported by the device.

The default depends on the device type. It is *n* if the device does not require a separate byte position for the attribute. Otherwise, the default is **y**.

SYSLIT = *ltseq*|*ltname*|*time*|*date1*|*date2*|*date3*|*date4*|*lpageno*|*ltmsg*
Identifies the field as a system literal.

SYSMSG = **n** | **y**
y specifies that the field is a system message field.

n specifies that the field is not a system message field. The default is **n**.

The extended tags DGIAID, DGIPFK, and DGIMFLD have no corresponding standard CSP/AD external source format tags. They are used to represent the information that is defined by the PFK parameter of the DEV utility control statement in IMS/MFS. In SDF II, this information is kept in an IMS/MFS AID (attention identifier) table.

Please note that these tags are not available with the initial shipment of SDF II Release 3. They are made available by design-APAR PN17032.

DGIAID
This tag defines an AID table.

AIDNAME = *table_name*
Identifies the AID table.

DATE = *date*
Specifies the date when the AID table was last modified.

TIME = *time*
Specifies the time when the AID table was last modified.

The format rules for the attributes are the same as for their equivalents in the MAP or MAPG tags.

One DGIAID plus one or more DGIPFK form one AID table definition. AID tables can be imported stand-alone or can be included in the map structure. If an AID table is imported within the map structure, the tags for the AID table must be before the associated tags for the map.

DGIPFK
This tag defines one PF key in an AID table.

KEY = *number*
Identifies the number of the PF key to be defined. It must be a number in the range of 1 through 24.

FUNCTION = *function*
Specifies the control function to be assigned to the PF key. These keywords can be specified:

NEXTPP	Page advance
NEXTMSG	Message advance
NEXTMSGP	Message advance protect
NEXTLP	Next logical page
ENDMPPI	End multiple page input

LITLEN = *length*
Specifies the length of the literal. It must be a number in the range of 1 through 256. If DBCS=Y is specified, it must be an even number.

DBCS = **n** | **y**
y specifies that the literal must be DBCS.

n specifies that the literal may be EBCDIC or mixed DBCS.

literal
Is the literal associated with the PF key.

- For DBCS=n, it may be EBCDIC or mixed DBCS.
- For DBCS=y, it must be DBCS. It must start with SO and end with SI without any other SO or SI between. The length of the literal is counted without the SO/SI symbols, which will be removed by SDF II.
- If the value specified for LITLEN is greater than the length of the literal, the literal will be padded with blanks up to the specified length.
- The maximum length of the literal is 256 characters.

DGIMFLD
This tag defines a PFK receiver field and relates it to an AID table. The field name corresponds to the name of the input field defined by the PFK parameter of the DEV utility control statement in IMS/MFS.

NAME = *field_name*
Identifies the name of the field. For the name the same rules apply as for the NAME attribute of the VFIELD tag.

BYTES = *field_length*
Specifies the field length in bytes. For PFK receiver fields, it must be a number in the range of 1 through 256.

AIDNAME = *aid_table_name*
Relates the field to an AID table from which the program function key literal or control function data is taken. The field will be added as PFK receiver field to the SDF II input data structure only.

Only one PFK receiver field can be defined in a map.

Restrictions

Within one map or map group structure, standard and extended tags must not be mixed.

The CSP/AD import utility does not accept extended external source format input.

If more than one device name is specified in the DEVICES attribute, import searches the device table for the smallest device list that contains these devices as a subset. If no device list is found, import for the object stops.

When a panel is created in a library, a panel group is created in the same library if:

- A panel group with the name specified for the GRPNAME attribute of the map structure does not already exist in the library.
- The name specified for the GRPNAME attribute is not the default name (DGIGRP).

If a panel group with the name specified for GRPNAME exists in the same library as the panel, the panel name will be added to the panel list regardless of the target system of the panel group.

If a panel group is renamed, all subsequently created panels for this group are added to the renamed group.

Importing CSP/AD export data sets

Use the import utility to create panels and panel groups from maps and map groups originally created with CSP/AD.

The import utility uses a CSP/AD export data set as input and creates SDF II objects.

Refer to “Importing external source format” on page 122 for importing CSP/AD external source format or extended external source format.

Here is how to create an export data set:

1. Use the export utility of CSP/AD to export the map set or the map from the MSL to the ALF data set.
2. Use the CSP's ALF utility to create an export data set. The ALF utility is a CSP application named UTILALF.ALFUTIL that runs under CSP/AE. Use the option PUT of the ALF utility.

You can concatenate several CSP/AD export data sets into one input data set.

For CSP/AD, a device type for map and map set import is defined by a unique 2-byte string. When you import from CSP/AD, define for each device type in any map or map group specification, an SDF II device that has the same name as the CSP/AD device name. This will be the device type of the created SDF II panel or panel group.

If a CSP/AD map or map group contains a specification that has more than one device type, the device list in the SDF II device table must contain all the device types of that specification.

SDF II scans the device table for each device type in the specification. It then scans for a device list that contains only these device types. The first device list that fulfills the condition will be the device type of the created SDF II panel or panel group.

Restrictions

If a format contains array elements in a random sequence, that is, if they are ordered neither horizontally nor vertically, the import utility reorders the array elements into a horizontally ordered array and displays a message.

Import of CSP/AD maps that have the same name, but belong to different map groups, creates duplicate names in SDF II. These duplicate names are processed as defined in the OPTION field of the Specify Import Utility Parameters panel.

The original names are retained in the description and as the generation names of any imported object.

Field starter characters and the spacer character can be changed by the import utility.

Importing objects from SDF/CICS

Use the import utility to create panels, panel groups, and partition sets from maps, map sets, and partition sets originally created using SDF/CICS.

The import uses an SDF/CICS dump data set as the source from which it creates SDF II objects.

Use the SDF/CICS unload utility with format DUMP to create a dump data set. You can concatenate several dump data sets into one input data set.

Use of the SDF II device table

For SDF/CICS, each device type has an associated SDF/CICS code. When you import objects from SDF/CICS, the device table is searched for an SDF II device with a matching SDF/CICS code.

The first device that has a matching code in the SDF II device table is the device type of the created SDF II panel or panel group.

If an SDF/CICS map or map set contains a specification that has more than one device type, the device list in the SDF II device table must contain all the device types of that specification.

SDF II scans the device table for each device type of the specification. It then scans for a device list that contains only these device types. The first device list fulfilling the condition is the device type of the created SDF II panel or panel group.

When you import an SDF/CICS partition set, an SDF II device with the same name as the partition set device name must exist. This will be the device type of the created SDF II partition set.

To update the device table, use the device table editor, which you can access in the SDF II customization dialog.

Restrictions

If a format contains array elements in a random sequence (that is, if they are ordered neither horizontally nor vertically), the import utility reorders the array elements into a horizontally ordered array and displays a message. Arrays of structures are imported as repeat formats. Arrays of structures that cannot be mapped to a repeat format are ignored.

Import of SDF/CICS maps that have the same name, but belong to different map sets, create duplicate names in SDF II. These duplicate names are processed as defined in the OPTION field of the Specify Import Utility Parameters panel.

The original names are retained in the description and as the generation names of any imported object.

If a partition set is defined with more than one suffix with the same partition set device, only the first specification is imported into SDF II.

Attributes associated with fields of length zero in SDF/CICS are normally not imported into SDF II.

Chapter 7. Migrating from one SDF II release to the next

The invocation EXEC generator

The invocation exec (DGIIXINV alias SDF2INV) can now be created in a customized form by generating it from an online customization dialog. Always use this dialog (DGIIXIN1) — do not modify DGIIXINV manually, This is described in “Generating the invocation routine” on page 5.

This invocation exec generator can also be used to update an existing customized version of the Release 4 invocation exec to a later level. This will not help you in migrating from Release 3 to Release 4, but it will ease migrating from Release 4 to any following release, and it will help in applying service that affects the invocation routine.

Migrating to Release 4

After installing a new release of a product, some effort is necessary to customize it, such that it behaves and looks like the previous release.

This chapter aims to help you in migrating from SDF II Release 3 to Release 4.

It identifies “customizable” parts that you can take unchanged from Release 3.

It also informs about changes in SDF II that may require definitions that now differ from those in Release 3.

Profile

The SDF II profile contains user-defined data for many settings in SDF II. It is usually located in *userid*`ISPF.ISPPR:pv,xx(DGIPROF)`, where *xx* is a language suffix:

EU English
JN Japanese
DU German
DS Swiss German
EP Spanish

A customized profile from Release 3 can be shared between Release 3 and Release 4.

The SDF II profile contains values for parameters such as

- selected target system
- overall editing defaults
- permanent window customizations
- print page size definitions
- some ISPF parameters
- parameters for Identify object panels (DGIxE00)
- List Objects parameters
- List Prototypes parameters
- parameters for all target system generations

- parameters in all utilities (print, import, ...)
- scroll amounts on window panels
- object library assignments (from specify libraries dialog)

Notes:

1. PF key assignments defined for SDF II Release 3 are also stored in the profile. These settings will not be used in Release 4 because for Release 4 each SDF II panel is associated with a keylist. If you want to customize a keylist, refer to “Changing ISPF keylist” on page 109.
2. A profile from Release 1 or Release 2 cannot be used in Release 3 or Release 4.

Device type table

Device type tables DGIKFDT and DGIKF10 in the SDF II table input library (if not otherwise customized SDF2.V1R3M0.SDGITLIB) can be taken from Release 3 unchanged.

Panels and messages

Data entry panels, help panels, OLR panels and messages have changed a lot in Release 4. If you have customized any of these, repeat the customization!

Customizations with SDF II Translation dialog

Customized operator input, column headings, panel commands, line commands, and panel text from Release 3 cannot be used.

Emphasis classes and CUA types

Emphasis class definitions are stored in table DGIKF30 on SDF II's table input library. They have not changed between Release 3 and Release 4.

However, the CUA table has changed:

- CUA type PS has been added:

```
CUA  Pro Attributes --------- Comment --------------------
PS   PR  BR TU                Point and Shoot
```

- The attribute UNDERLINED has been added to the CUA types CEF and NEF.

You can use your Release 3 CUA types table (DGIKF40) and may choose to update it according to the changes in Release 4.

Skeletons

Skeletons have not changed for Release 4. You may use your customized Release 3 skeleton files from both the language-independent library SDF2.V1R3M0.SDGISK, and the NLS-dependent library SDF2.V1R3M0.SDGISK*xx* without any changes (*xx* is the language suffix).

Upper case and lower case translation

Uppercase/lowercase translation tables (DGIYTUC and DGIYUTLC in SDF2.V1R3M0.SDGISAM) can be used from Release 3.

CLIST versions of SDF II routines dropped

The SDF II routines that previously were available both as a CLIST and as REXX execs are available only as REXX execs in Release 4.

Prototype users who have used the CLIST version of the prototype interpretation routine DGIWX20 should check their prototype definitions for ACTIONS and CONDITIONS; these may contain statements in CLIST syntax that cannot be interpreted as REXX statements.

NLS-independent yes/no selections

The user interface of SDF II now uses check box fields for fields that required a response of **Y** or **N** (for yes or no). When using NLS version other than English, the corresponding translations for Y and N had to be used.

With Release 4, these fields are NLS-independent. When setting up a request file for batch processing, the following assignments must be used:

`selected` or /	equivalent to Y
`unselected` or blank	equivalent to N

Samples

Check the following two modules, as they have changed in Release 4 to accommodate flexible data set names:

DGIIEDT
DGILXLI

Appendix A. The device table

This appendix lists devices supported by SDF II. The column headings in the table are abbreviated. The following list expands them:

Column heading	Meaning
NAM	Device name
DVL	Indication for device list (y=DL, n, or blank=no DL)
SDF	SDF/CICS device code
BMS	CICS/BMS device suffix
GDD	GDDM-IMD device class
MFS	IMS/MFS device type
CSP	CSP/AD device name
DEP	Depth in characters
WID	Width in characters
DPP	Depth in pels
WIP	Width in pels
PDB	Maximum partitions buffer size
PSB	Maximum scroll buffer size
CVM	Character cell size vertical, minimum
CVX	Character cell size vertical, maximum
CHM	Character cell size horizontal, minimum
CHX	Character cell size horizontal, maximum
PFK	Number of PF keys
NPA	Number of partitions
PGH	Number of horizontal gaps between partitions
IBM	An **I** indicates: Device entry provided by IBM
EXS	An **E** indicates: Extendable device size
INP	An **I** indicates: This is an input device
OUT	An **O** indicates: This is an output device
ADJ	An **A** indicates: Adjacent fields possible
NUM	An **N** indicates: Numeric lock feature
PEN	A **P** indicates: Light-pen available
COL	A **C** indicates: Color feature
HIL	An **H** indicates: Extended highlighting feature
PSS	A **P** indicates: Programmed symbol set feature
DBC	A **D** indicates: DBCS support

VAL	A **V** indicates: Field validation feature
LIN	An **L** indicates: Field outlining feature
MIX	An **M** indicates: Mixed EBCDIC/DBCS
TRA	A **T** indicates: Transparency feature
CAT	A **C** indicates: Character attributes
TAB	A **T** indicates: Tabulators
LDC	An **L** indicates: Logical device code
OBF	An **O** indicates: Outboard formatting

```
                                                                                                                IEIOANPCHPDVLMTCTLO
                                                                                                                BXNUDUEOISBAIIRAADB
NAM      DVL SDF BMS GDD MFS      CSP       DEP WID DPP WIP PDB PSB CVMCVXCHMCHXPFKNPAPGHPGV MSPTJMNLLSCLNXATBCF
-------------------------------------------------------------------------------------------------------------------------------
3270         F3          3270              24  80                              24          I IO   CHPDVLM
3270-1       F1  L       3270-A5           12  40                              24          I IO
3270-2       F2  M       3270-A2           24  80                              24          I IO
ALL          F4          3270-A2          240 240                              24          I IO   CHPDVLM
CRLP         F5  A                         12  80                                          I IO
TAPE         F6  B                         12  80                                          I IO
DISK         F7  C                         12  80                                          I IO
TWX          F8  D                         12  80                                          I IO
1050         F9  E                         12  80                                          I IO
2740         C1  F                         12  80                                          I IO
2741         C2  G                         12  80                                          I IO
2770         C3  I                         12  80                                          I IO
2780         C4  J                         12  80                                          I IO
3780         C5  K                         12  80                                          I IO
INTLU        C6  P                         12  80                                          I IO            TL
3767         C7  P   R1          3767      66 132                                          IEIO            TL
3770I        C8  P                         12  80                                          I IO            TL
SCS          C9  P                         24  80                                          I IO            TL
6670         D1  P                         50  80                                          I IO            TL
2980         D2  Q                         12  40                                          I IO
2980-4       D3  R                         12  40                                          I IO
3601         D4  U                          1  40                                          I IO             L
3653         D5  V                          6  30                                          I IO
3650UP       D6  W                          3  80                                          I IO
3650/32      D7  X                         23  80                                          I IO              O
BCHLU        D8  Y                         12  80                                          I IO            TL
3770B        D9  Y                         12  80                                          I IO            TL
274X                     274X              55 132                                          I IOA
3270,1                   3270,1            12  40                              24          I IO   CHP V
3270,2                   3270,2            24  80                              24          I IO   CHPDVLM
3270-A1                  3270-A1           12  80                              24          I IO   CHP V
3270-A2                  3270-A2           24  80                              24          I IO   CHPDVLM
3270-A3                  3270-A3           32  80                              24          I IO   CHP V
3270-A4                  3270-A4           43  80                              24          I IO   CHP V
3270-A5                  3270-A5           12  40                              24          I IO   CHP V
3270-A6                  3270-A6            6  40                              24          I IO   CHP V
3270-A7                  3270-A7           27 132                              24          I IO   CHP V
3270-A8                  3270-A8           62 160 751 960          12 31  6 12 24 16       IEIO   CHP V
3270P                    3270P             55 132                                          I  OA
3270P,1                  3270P,1           55 132                                          I  OA
3270P,2                  3270P,2           55 132                                          I  OA
FIN                      FIN               24  80                                          IEI A
FIDS                     FIDS               6  40                                          I  OA
FIDS3                    FIDS3             12  40                                          I  OA
FIDS4                    FIDS4             16  64                                          I  OA
FIDS7                    FIDS7             24  80                                          I  OA
FIJP                     FIJP              55  80                                          I  OA
FIPB                     FIPB              55 100                                          I  OA
FIFP                     FIFP              55  80                                          I  OA
FIFP132                  FIFP              55 132                                          I  OA
3600                     3600              24  80                                          IEI A
36DS                     36DS               6  40                                          I  OA
36DS3                    36DS3             12  40                                          I  OA
36DS4                    36DS4             16  64                                          I  OA
36DS7                    36DS7             24  80                                          I  OA
36JP                     36JP              55  80                                          I  OA
36PB                     36PB              55 100                                          I  OA
36FP                     36FP              55  80                                          I  OA
36FP132                  36FP              55 132                                          I  OA
SCS1                     SCS1              55 132                                          I IOA     D LM
SCS2                     SCS2              55  80                                          I IOA
```

Figure 71 (Part 1 of 3). The device table

```
                                                                                                                IEIOANPCHPDVLMTCTLO
                                                                                                                BXNUDUEOISBAIIRAADB
NAM     DVL SDF BMS GDD MFS     CSP      DEP WID DPP WIP PDB PSB CVMCVXCHMCHXPFKNPAPGHPGV MSPTJMNLLSCLNXATBCF
-------------------------------------------------------------------------------------------------------------------------------
DPM-A1                  DPM-A1                                                          I IO
DPM-A2                  DPM-A2                                                          I IO
DPM-A3                  DPM-A3                                                          I IO
DPM-A4                  DPM-A4                                                          I IO
DPM-A5                  DPM-A5                                                          I IO
DPM-A6                  DPM-A6                                                          I IO
DPM-A7                  DPM-A7                                                          I IO
DPM-A8                  DPM-A8                                                          I IO
DPM-A9                  DPM-A9                                                          I IO
DPM-A10                 DPM-A10                                                         I IO
DPM-A11                 DPM-A11                                                         I IO
DPM-A12                 DPM-A12                                                         I IO
DPM-A13                 DPM-A13                                                         I IO
DPM-A14                 DPM-A14                                                         I IO
DPM-A15                 DPM-A15                                                         I IO
DPM-B1                  DPM-B1                                                          I IO
DPM-B2                  DPM-B2                                                          I IO
DPM-B3                  DPM-B3                                                          I IO
DPM-B4                  DPM-B4                                                          I IO
DPM-B5                  DPM-B5                                                          I IO
DPM-B6                  DPM-B6                                                          I IO
DPM-B7                  DPM-B7                                                          I IO
DPM-B8                  DPM-B8                                                          I IO
DPM-B9                  DPM-B9                                                          I IO
DPM-B10                 DPM-B10                                                         I IO
DPM-B11                 DPM-B11                                                         I IO
DPM-B12                 DPM-B12                                                         I IO
DPM-B13                 DPM-B13                                                         I IO
DPM-B14                 DPM-B14                                                         I IO
DPM-B15                 DPM-B15                                                         I IO
8775-1C         C4      8775-1C   12  80         4004779              24  8  2    IEIO   CHP V
8775-1D         D4      8775-1D   12  80         4004779              24  8  2    IEIO   CHP V
8775-2C         C5      8775-2C   24  80         4004779              24  8  2    IEIO   CHP V
8775-2D         D5      8775-2D   24  80         4004779              24  8  2    IEIO   CHP V
8775-3C         C6      8775-3C   32  80         4004779              24  8  2    IEIO   CHP V
8775-3D         D6      8775-3D   32  80         4004779              24  8  2    IEIO   CHP V
8775-4C         C7      8775-4C   43  80         4004779              24  8  2    IEIO   CHP V
8775-4D         D7      8775-4D   43  80         4004779              24  8  2    IEIO   CHP V
3277-1          A2      3277-1    12  40                              24          I IO
3643-2          A1      3643-2     6  40                                          I IO   CHP V
3643-4          A3      3643-4    16  64                                          I IO   CHP V
ANY-1D          D4      ANY-1D    12  80                              24  8  2    IEIO   CHP V
ANY-2D          D5      ANY-2D    24  80                              24  8  2    IEIO   CHP V
ANY-3D          D6      ANY-3D    32  80                              24  8  2    IEIO   CHP V
ANY-4D          D7      ANY-4D    43  80                              24  8  2    IEIO   CHP V
ANY-5D          D8      ANY-5D    27 132                              24  8  2    IEIO   CHP V
ANY-D           D9      ANY-D     62 160 751 960          12 31  6 12 24 16       IEIO    HP
PRINTER         P1      PRINTER   66 132                                          IE O
PRINT-B         Q1      PRINT-B   66 132                                          IE O
5550D           K5      5550D     24  80                              24          I IO   CH D LM
A1              A1      3643-2     6  40                              24          I IO
A2              A2      3277-1    12  40                              24          I IO
A3              A3      3643-4    16  64                              24          I IO
A4              A4      3278-1    12  80                              24          I IO
A5              A5      3278-2    24  80                              24          I IO
A6              A6      3278-3    32  80                              24          I IO
A7              A7      3278-4    43  80                              24          I IO
A8              A8      3278-5    27 132                              24          I IO
B4              B4      3278-1B   12  80                              24          I IO   CHP V
B5              B5      3278-2B   24  80                              24          I IO   CHP V
B6              B6      3278-3B   32  80                              24          I IO   CHP V
B7              B7      3278-4B   43  80                              24          I IO   CHP V
B8              B8      3278-5B   27 132                              24          I IO   CHP V
```

Figure 71 (Part 2 of 3). The device table

```
                                                                                                                IEIOANPCHPDVLMTCTLO
                                                                                                                BXNUDUEOISBAIIRAADB
NAM      DVL SDF BMS GDD MFS      CSP       DEP WID DPP WIP PDB PSB CVMCVXCHMCHXPFKNPAPGHPGV MSPTJMNLLSCLNXATBCF
-------------------------------------------------------------------------------------------------------------------------
C4                  C4           8775-1C   12  80                                 24          I IO   CHP V
C5                  C5           8775-2C   24  80                                 24          I IO   CHP V
C6                  C6           8775-3C   32  80                                 24          I IO   CHP V
C7                  C7           8775-4C   43  80                                 24          I IO   CHP V
D0                  D0                      6  20                                 24          I IO   CHP V
D1                  D1                      6  40                                 24          I IO   CHP V
D2                  D2                     12  40                                 24          I IO   CHP V
D3                  D3                     16  64                                 24          I IO   CHP V
D4                  D4           8775-1D   12  80                                 24          I IO   CHP V
D5                  D5           8775-2D   24  80                                 24          I IO   CHP V
D6                  D6           8775-3D   32  80                                 24          I IO   CHP V
D7                  D7           8775-4D   43  80                                 24          I IO   CHP V
D8                  D8                     27 132                                 24          I IO   CHP V
D9                  D9           ANY-D     62 160                                 24          I IO   CHP V
K5                  K5           5550D     24  80                                 24          I IO   CH D LM
K6                  K6                     32  80                                 24          I IO   CH D LM
K7                  K7                     43  80                                 24          I IO   CH D LM
P1                  P1           PRINTER   66 132                                             I  O
Q1                  Q1           PRINT-B   66 132                                             I  O
R1                  R1           3767      66 132                                 24          I IO            TL
V1                  V1           5550P     66 158                                             I IO      D LM
ISPFDEV             D5  3270-A2  3278-2B   24  80                                 24          IEIO   CH D LM
3278-1      01  1   A4  3270-A1  3278-1    12  80                                 24          I IO   CHP V
3278-2      02  2   A5  3270-A2  3278-2    24  80                                 24          I IO   CHP V
3278-3      03  3   A6  3270-A3  3278-3    32  80                                 24          I IO   CHP V
3278-4      04  4   A7  3270-A4  3278-4    43  80                                 24          I IO   CHP V
3278-5      05  5   A8  3270-A7  3278-5    27 132                                 24          I IO   CHP V
3278-1B     06  1   B4  3270-A1  3278-1B   12  80                                 24          I IO   CHP V
3278-2B     07  2   B5  3270-A2  3278-2B   24  80                                 24          I IO   CHP V
3278-3B     08  3   B6  3270-A3  3278-3B   32  80                                 24          I IO   CHP V
3278-4B     09  4   B7  3270-A4  3278-4B   43  80                                 24          I IO   CHP V
3278-5B     0A  5   B8  3270-A7  3278-5B   27 132                                 24          I IO   CHP V
3279-2B     17  2   D5  3270-A2  3278-2B   24  80                                 24          I IO   CHP V
3279-3B     18  3   D6  3270-A3  3278-3B   32  80                                 24          I IO   CHP V
3290        0D  9       3270-A8            62 160 751 960          12 31  6 12 24 16       IEIO    HP
8775        0C  7       3270-A4            43  80         4004779                 24  8  2    IEIO   CHP V
5550        1C  8   K5  3270-A2  5550D     24  80                                 24          I IO   CH D LM
3278-52     0B  8   K5  3270-A2  5550D     24  80                                 24          I IO      D
3278-K1     1B  8       3270-A2            24  80                                 24          I IO   CH D LM
5550P       2C  0   V1  SCS1     5550P    127 158                                             IE OA     D LM
3283-52             V1           5550P    127 158                                             IE O      D
3283-K1     2B  0                        127 158                                             I  O      D LM
3179            2       3270-A2            24  80                                 24          I IO   CHP V
3179-G          3       3270-A3            32  80                                 24          I IO   CHP V
3180            5       3270-A2            27 132           7680                  24  1       IEIO   CHP V
```

Figure 71 (Part 3 of 3). The device table

Appendix B. SDF II variables

The following table is a list of all the variables that are used in SDF II The table is sorted by name and will help you to locate the detailed description of the variable's meaning and use.

Figure 72 (Page 1 of 7). Sorted list of SDF II variables

Name	Used in	Description	Reference
DGIII*	Invocation exits		Figure 1 on page 13
ICVUEGTY	Generation exit	Generation type	Figure 52 on page 73
ICVUELAN	Generation exit	Programming language	Figure 52 on page 73
ICVUELIB	Generation exit	Library ID	Figure 52 on page 73
ICVUENAM	Generation exit	Object name	Figure 52 on page 73
ICVUEODD	Generation exit	Output file ddname	Figure 52 on page 73
ICVUEOLN	Generation exit	Output file data set name	Figure 52 on page 73
ICVUEOMN	Generation exit	Output member name	Figure 52 on page 73
ICVUETSY	Generation exit	Target system	Figure 52 on page 73
ICVUETYP	Generation exit	Object type	Figure 52 on page 73
ICV10ADI	SDF II batch	input field variable	Figure 20 on page 53
ICV10CDT	SDF II batch	input field variable	Figure 20 on page 53
ICV10COD	SDF II batch	input field variable	Figure 20 on page 53
ICV10COL	SDF II batch	input field variable	Figure 20 on page 53
ICV10CUE	SDF II batch	input field variable	Figure 20 on page 53
ICV10DAL	SDF II batch	input field variable	Figure 20 on page 53
ICV10DGR	SDF II batch	input field variable	Figure 20 on page 53
ICV10DLA	SDF II batch	input field variable	Figure 20 on page 53
ICV10DOD	SDF II batch	input field variable	Figure 20 on page 53
ICV10DOL	SDF II batch	input field variable	Figure 20 on page 53
ICV10DTR	SDF II batch	input field variable	Figure 20 on page 53
ICV10DUE	SDF II batch	input field variable	Figure 20 on page 53
ICV10GCB	SDF II batch	input field variable	Figure 20 on page 53
ICV10GDS	SDF II batch	input field variable	Figure 20 on page 53
ICV12COD	SDF II batch	input field variable	Figure 22 on page 55
ICV12COL	SDF II batch	input field variable	Figure 22 on page 55
ICV12CUE	SDF II batch	input field variable	Figure 22 on page 55
ICV12DGR	SDF II batch	input field variable	Figure 22 on page 55
ICV12DLA	SDF II batch	input field variable	Figure 22 on page 55
ICV12DOD	SDF II batch	input field variable	Figure 22 on page 55
ICV12DOL	SDF II batch	input field variable	Figure 22 on page 55
ICV12DUE	SDF II batch	input field variable	Figure 22 on page 55
ICV12GCB	SDF II batch	input field variable	Figure 22 on page 55

Figure 72 (Page 2 of 7). Sorted list of SDF II variables

Name	Used in	Description	Reference
ICV12GDS	SDF II batch	input field variable	Figure 22 on page 55
ICV12IOT	SDF II batch	input field variable	Figure 22 on page 55
ICV12IPS	SDF II batch	input field variable	Figure 22 on page 55
ICV13COD	SDF II batch	input field variable	Figure 22 on page 55
ICV13COL	SDF II batch	input field variable	Figure 22 on page 55
ICV13CUE	SDF II batch	input field variable	Figure 22 on page 55
ICV13DGR	SDF II batch	input field variable	Figure 22 on page 55
ICV13DLA	SDF II batch	input field variable	Figure 22 on page 55
ICV13DOD	SDF II batch	input field variable	Figure 22 on page 55
ICV13DOL	SDF II batch	input field variable	Figure 22 on page 55
ICV13DTR	SDF II batch	input field variable	Figure 22 on page 55
ICV13DUE	SDF II batch	input field variable	Figure 22 on page 55
ICV13GCP	SDF II batch	input field variable	Figure 22 on page 55
ICV13GDS	SDF II batch	input field variable	Figure 22 on page 55
ICV14COD	SDF II batch	input field variable	Figure 26 on page 57
ICV14COL	SDF II batch	input field variable	Figure 26 on page 57
ICV14CUE	SDF II batch	input field variable	Figure 26 on page 57
ICV14DGR	SDF II batch	input field variable	Figure 26 on page 57
ICV14DLA	SDF II batch	input field variable	Figure 26 on page 57
ICV15CDT	SDF II batch	input field variable	Figure 28 on page 58
ICV15COD	SDF II batch	input field variable	Figure 28 on page 58
ICV15COL	SDF II batch	input field variable	Figure 28 on page 58
ICV15CPG	SDF II batch	input field variable	Figure 28 on page 58
ICV15CSE	SDF II batch	input field variable	Figure 28 on page 58
ICV15CUE	SDF II batch	input field variable	Figure 28 on page 58
ICV20COD	SDF II batch	input field variable	Figure 30 on page 59
ICV20COL	SDF II batch	input field variable	Figure 30 on page 59
ICV20CUE	SDF II batch	input field variable	Figure 30 on page 59
ICV20DTY	SDF II batch	input field variable	Figure 30 on page 59
IIVBOML	BOM table	Library ID	Figure 53 on page 74
IIVBOMN	BOM table	Referenced object name	Figure 53 on page 74
IIVBOMT	BOM table	Referenced object type	Figure 53 on page 74
IIVDEV	Table DGIIFTAB	Device type	Figure 13 on page 39
	SDF II batch	input field variable	Figure 38 on page 63

Figure 72 (Page 3 of 7). Sorted list of SDF II variables

Name	Used in	Description	Reference
IIVLIB	Table DGIIFTAB	Library ID	Figure 13 on page 39
	SDF II batch	input field variable	Figure 19 on page 52
	SDF II batch	input field variable	Figure 32 on page 60
	SDF II batch	input field variable	Figure 34 on page 61
	SDF II batch	input field variable	Figure 38 on page 63
IIVMOD	Table DGIIFTAB	EDIT: Editor option GENERATE: Target system PRINT: SDF II output destination	Figure 13 on page 39
	SDF II batch	input field variable	Figure 19 on page 52
	SDF II batch	input field variable	Figure 36 on page 62
IIVNAM	Table DGIIFTAB	Object name	Figure 13 on page 39
	SDF II batch	input field variable	Figure 19 on page 52
	SDF II batch	input field variable	Figure 32 on page 60
	SDF II batch	input field variable	Figure 34 on page 61
	SDF II batch	input field variable	Figure 36 on page 62
	SDF II batch	input field variable	Figure 38 on page 63
IIVOLIB	Table DGIIFTAB	EDIT/COPY: Library ID CONSTRUCT: ISPF panel description	Figure 13 on page 39
	SDF II batch	input field variable	Figure 39 on page 64
IIVONAM	Table DGIIFTAB	EDIT/COPY: Object name CONSTRUCT: ISPF table name	Figure 13 on page 39
IIVPRO	Table DGIIFTAB	Panel disposition	Figure 13 on page 39
IIVREQ	Table DGIIFTAB	Request	Figure 13 on page 39
IIVROW	Table DGIIFTAB	Row type	Figure 13 on page 39
IIVTYP	Table DGIIFTAB	Object type	Figure 13 on page 39
	SDF II batch	input field variable	Figure 19 on page 52
	SDF II batch	input field variable	Figure 32 on page 60
	SDF II batch	input field variable	Figure 34 on page 61
ILVUEASK	Exit DGILXLI out	Access key	Figure 5 on page 29
ILVUEATC	Exit DGILXLI out	Authorization code	Figure 5 on page 29
ILVUECGC	Exit DGILXLI out	Change code	Figure 5 on page 29
ILVUEGN1	Exit DGILXLI out	Group name of lowest SCLM group	Figure 5 on page 29
ILVUELIB	Exit DGILXLI in	Library specification	Figure 5 on page 29
	Exit DGILXOL in	Library specification	Figure 6 on page 30
	Exit DGILXOD in	Library specification	Figure 9 on page 32
	Exit DGILXOR	Library specification	Figure 10 on page 33
	Exit DGILXIO	Library specification	Figure 11 on page 34

Figure 72 (Page 4 of 7). Sorted list of SDF II variables

Name	Used in	Description	Reference
ILVUELID	Exit DGILXOL i/o	Library ID	Figure 6 on page 30
	Exit DGILXOD i/o	Library ID	Figure 9 on page 32
	Exit DGILXOR	Library ID	Figure 10 on page 33
	Exit DGILXIO	Library ID	Figure 11 on page 34
ILVUELUD	Exit DGILXLI i/o	Communication area	Figure 5 on page 29
	Exit DGILXOL i/o	Communication area	Figure 6 on page 30
	Exit DGILXOD i/o	Communication area	Figure 9 on page 32
	Exit DGILXOR	Communication area	Figure 10 on page 33
	Exit DGILXIO	Communication area	Figure 11 on page 34
ILVUEMSG	Exit DGILXLI out	Message ID	Figure 5 on page 29
	Exit DGILXOL out	Message ID	Figure 6 on page 30
	Exit DGILXOD out	Message ID	Figure 9 on page 32
	Exit DGILXOR	Message ID	Figure 10 on page 33
	Exit DGILXIO	Message ID	Figure 11 on page 34
ILVUENAM	Exit DGILXOL in	Requested object names	Figure 6 on page 30
	Exit DGILXOD in	Object name	Figure 9 on page 32
	Exit DGILXOR	Object name	Figure 10 on page 33
	Exit DGILXIO	Object name	Figure 11 on page 34
ILVUENNM	Exit DGILXOR	New object name	Figure 10 on page 33
ILVUEOPM	Exit DGILXOL i/o	Library status	Figure 6 on page 30
	Exit DGILXOD i/o	Library status	Figure 9 on page 32
	Exit DGILXOR	Library status	Figure 10 on page 33
	Exit DGILXIO	Library status	Figure 11 on page 34
ILVUEORG	Exit DGILXOL i/o	Data set organization	Figure 6 on page 30
	Exit DGILXOD i/o	Data set organization	Figure 9 on page 32
	Exit DGILXOR	Data set organization	Figure 10 on page 33
	Exit DGILXIO	Data set organization	Figure 11 on page 34
ILVUEPNM	Exit DGILXLI out	SCLM project name	Figure 5 on page 29
ILVUEPWD	Exit DGILXLI in	Password	Figure 5 on page 29
	Exit DGILXOL i/o	Password	Figure 6 on page 30
	Exit DGILXOD i/o	Password	Figure 9 on page 32
	Exit DGILXOR	Password	Figure 10 on page 33
	Exit DGILXIO	Password	Figure 11 on page 34
ILVUETYP	Exit DGILXOL in	Requested object type	Figure 6 on page 30
	Exit DGILXOD in	Requested object type	Figure 9 on page 32
	Exit DGILXOR	Requested object type	Figure 10 on page 33
	Exit DGILXIO	Requested object type	Figure 11 on page 34
IPVENFFO	Copy utility out	Field format of item	Figure 16 on page 47

Figure 72 (Page 5 of 7). Sorted list of SDF II variables

Name	Used in	Description	Reference
IPVENTXT	Copy utility out	Text associated with item	Figure 16 on page 47
IPVEOARX	Copy utility	Array index	Figure 16 on page 47
IPVEOCOL	Copy utility	Horizontal position of the item	Figure 16 on page 47
IPVEOFFO	Copy utility	Field format of the item	Figure 16 on page 47
IPVEOINC	Copy utility	Include panel name	Figure 16 on page 47
IPVEOLEN	Copy utility	Item length	Figure 16 on page 47
IPVEOLIN	Copy utility	Vertical position of the item	Figure 16 on page 47
IPVEOLIX	Copy utility	Repeat format index	Figure 16 on page 47
IPVEONAM	Copy utility	Field name	Figure 16 on page 47
IPVEOTXT	Copy utility	Text associated with item	Figure 16 on page 47
IPVFODEP	Copy utility	Format depth	Figure 16 on page 47
IPVFODVT	Copy utility	Device type of format instance	Figure 16 on page 47
IPVFOFON	Copy utility	Format name	Figure 16 on page 47
IPVFOPNB	Copy utility	Number of physical pages	Figure 16 on page 47
IPVFOPNN	Copy utility	Index number of physical page	Figure 16 on page 47
IPVFOWID	Copy utility	Format width	Figure 16 on page 47
IUI10PAN	ISPF dialog	ISPF table	Figure 58 on page 99
IUI10PA1	ISPF dialog	ISPF table element containing attributes	Figure 58 on page 99
IUI10PA2	ISPF dialog	ISPF table element containing ruling attributes	Figure 58 on page 99
IUI10PLI	ISPF dialog	ISPF table element containing the panel lines	Figure 58 on page 99
IUVUEDES	Exit DGIUXRET out	Field description	Figure 48 on page 69
	Exit DGIUXEXP out	Field description	Figure 50 on page 70
IUVUEFFO	Exit DGIUXRET out	Field format	Figure 48 on page 69
	Exit DGIUXEXP out	Field format	Figure 50 on page 70
IUVUEINI	Exit DGIUXRET out	Field initial value	Figure 48 on page 69
	Exit DGIUXEXP out	Field initial value	Figure 50 on page 70
IUVUELEN	Exit DGIUXRET out	Field length	Figure 48 on page 69
	Exit DGIUXEXP out	Field length	Figure 50 on page 70
IUVUEMSG	Exit DGIUXRET out	Private message ID	Figure 48 on page 69
	Exit DGIUXEXP out	Private message ID	Figure 50 on page 70
IUVUEOCC	Exit DGIUXRET out	Occurrence number	Figure 48 on page 69
	Exit DGIUXEXP out	Occurrence number	Figure 50 on page 70
IUVUEPRO	Exit DGIUXRET out	Field prompt	Figure 48 on page 69
	Exit DGIUXEXP out	Field prompt	Figure 50 on page 70
IUVUERFD	Exit DGIUXRET in	Related field name	Figure 48 on page 69
	Exit DGIUXEXP in	Related field name	Figure 50 on page 70
IUVUNAM	Exit DGIUXEXP out	SDF II field name	Figure 50 on page 70
IUV10CNT	SDF II batch	input field variable	Figure 34 on page 61

Figure 72 (Page 6 of 7). Sorted list of SDF II variables

Name	Used in	Description	Reference
IUV10DSN	SDF II batch	input field variable	Figure 34 on page 61
IUV10FMT	SDF II batch	input field variable	Figure 34 on page 61
IUV10PID	SDF II batch	input field variable	Figure 34 on page 61
IUV10PDE	ISPF dialog	Panel depth	Figure 58 on page 99
IUV10PWI	ISPF dialog	Panel width	Figure 58 on page 99
IUV14ANM	ISPF dialog	Scrollable area name	Figure 58 on page 99
IUV14CMA	ISPF dialog	Selection criteria	Figure 58 on page 99
IUV14DSN	ISPF dialog	Data set name or blank	Figure 58 on page 99
IUV14FMT	ISPF dialog	Print format	Figure 58 on page 99
IUV14MEM	ISPF dialog	Member name or printer ID	Figure 58 on page 99
IUV14OSY	ISPF dialog	Operating system	Figure 58 on page 99
IUV14PDV	ISPF dialog	Device type	Figure 58 on page 99
IUV14PFN	ISPF dialog	Target system	Figure 58 on page 99
IUV14PLB	ISPF dialog	Object library	Figure 58 on page 99
IUV14PPN	ISPF dialog	Object name	Figure 58 on page 99
IUV14PPP	ISPF dialog	Physical page	Figure 58 on page 99
IUV14PTY	ISPF dialog	Object type	Figure 58 on page 99
IUV14TSY	ISPF dialog	Object target system	Figure 58 on page 99
IUV20CL	SDF II batch	input field variable	Figure 32 on page 60
IUV20EO	SDF II batch	input field variable	Figure 32 on page 60
IUV20SD	SDF II batch	input field variable	Figure 32 on page 60
IUV20SL	SDF II batch	input field variable	Figure 32 on page 60
IUV40DSN	SDF II batch	input field variable	Figure 36 on page 62
IUV5DES	Table DGIU5TAB	ISPF procedural section line or field description	Figure 13 on page 39
	SDF II batch	input field variable	Figure 39 on page 64
IUV5FFO	Table DGIU5TAB	Field format	Figure 13 on page 39
IUV5LEN	Table DGIU5TAB	Data field length	Figure 13 on page 39
	SDF II batch	input field variable	Figure 39 on page 64
IUV5NAM	Table DGIU5TAB	SDF II field name	Figure 13 on page 39
	SDF II batch	input field variable	Figure 39 on page 64
IUV5OCC	Table DGIU5TAB	Occurrence number or array dimension	Figure 13 on page 39
	SDF II batch	input field variable	Figure 39 on page 64
IUV5PRO	Table DGIU5TAB	Prompt text	Figure 13 on page 39
	SDF II batch	input field variable	Figure 39 on page 64
IUV5RFD	Table DGIU5TAB	Related field name	Figure 13 on page 39
	SDF II batch	input field variable	Figure 39 on page 64
IUV5TYP	Table DGIU5TAB	Panel element type	Figure 13 on page 39

Figure 72 (Page 7 of 7). Sorted list of SDF II variables

Name	Used in	Description	Reference
IUV5VER	Table DGIU5TAB	Direction	Figure 13 on page 39
	SDF II batch	input field variable	Figure 39 on page 64
IUV62FUN	Copy utility out	Function	Figure 16 on page 47
IUV62LIB	Copy utility out	Source library ID	Figure 16 on page 47
IUV62NAM	Copy utility out	Source object name	Figure 16 on page 47
IUV62NLI	Copy utility out	Target library ID	Figure 16 on page 47
IUV62NNA	Copy utility out	Target object name	Figure 16 on page 47
IUV62REQ	Copy utility out	Item type	Figure 16 on page 47
IUV62TYP	Copy utility out	Object type	Figure 16 on page 47

Appendix C. Diagnosing error situations

This appendix briefly summarizes the steps you can take when an error situation occurs.

Preparing an APAR

If the keyword search is unsuccessful or if you do not have access to the IBM Software Support Databases, you must prepare an authorized program analysis report (APAR). The first step in preparing an APAR is to contact your IBM Support Center.

Initiating an APAR

Contact your IBM Support Center for assistance either directly or through an IBM Program Support Representative. You should be prepared to supply the following information:

- Your operating system and release level
- Release and PTF level of SDF II
- The keyword set that was used to search the software support database
- A free form problem description

Gathering APAR documentation

You will be asked for information that describes the failure in SDF II. You may be asked to include some items from the following list together with your APAR:

- A failing test case, made as short as possible while still demonstrating the failure. If it is too long to allow easy keyboard entry from your written description, it should be on magnetic tape (unlabeled or standard labeled tape).

  It is best to include the actual SDF II object that showed the failure.
- A hard copy of job control statements or the CMS procedure for unloading the submitted machine-readable tape.
- An example of the failing output and of the expected output.
- Data sets:
  - ISPF list data set
  - ISPF log data set.
- Output from the SDF II trace facility (DGIPRINT).
- Current service level. The consolidated software inventory, containing the target zone or zones.
- A storage dump.
- Console output from the system on which the job was run, or a listing of the SYSLOG data set for the period of time spanning the failure.
- For an import utility failure:
  - The input to the failing utility
  - The imported objects in SDF II that fail or show the problem.

With this information, IBM Support Center personnel can try to reproduce the failure and observe the symptoms of the failure.

Note: The IBM Support Center personnel will assign you an APAR number.

Submitting an APAR

When you submit material for an APAR to IBM, ensure that it is all clearly identified and carefully labeled.

Submitting printed material as APAR documentation

Ensure that any printed material, including dumps, is clearly marked with the assigned APAR number in the top right-hand corner.

The SDF II trace facility

This appendix explains how to start and use the SDF II trace facility and how to interpret the trace listing.

The SDF II trace facility produces a chronological listing of SDF II services and indicates where in SDF II each event occurred. The listing also shows whether or not the event was successful.

Using the trace facility may affect the performance of SDF II because all the trace information is written to a file.

Starting the SDF II trace facility in an online environment

To start the SDF II trace facility when you are in an online environment, enter one of these trace commands:

trace all Starts the SDF II trace facility. This command produces a detailed list of services. The **trace all** command writes this list of SDF II services to DGIPRINT. (DGIPRINT is the file that contains print output formatted for the system printer.) This command is recommended for all cases except loops and large traces.

trace on Starts the SDF II trace facility and writes a less detailed list of services than **trace all.** The **trace on** command also writes this list of SDF II services to DGIPRINT. This command is recommended only for loops and large traces.

To stop the SDF II trace facility, enter:

trace off Stops tracing SDF II services.

Starting the SDF II trace facility in a batch environment

To start the SDF II trace facility in batch, add the trace option to the command you use to start SDF II. The syntax depends on the operating system:

SDF2INV REQUEST(SDF2X) [TRACE({OFF|ON|ALL})]

The trace options have the following meanings:

ALL Starts the SDF II trace facility. This option produces a detailed list of services. The ALL option writes this list of SDF II services to DGIPRINT. This option is recommended for all cases except loops and large traces.

ON Starts the SDF II trace facility, and writes a less detailed list of services than the ALL option. The ON option also writes this list of SDF II services to DGIPRINT. This option is recommended only for loops and large traces.

OFF Requests no tracing. This is the default.

The trace listing

You can print trace information on the system printer.

A trace entry is written whenever an event starts or stops. It appears as:

```
hh:mm:ss.ttt nnnnnnnn oooo mmmmmmmm xrrrr parameters
```

This is what the entries mean:

`hh:mm:ss.ttt` The time stamp.

`nnnnnnnn` The name of the module issuing the request.

`oooo` The hexadecimal offset of the instruction in the module that issued the request.

`mmmmmmmm` The name of the requested service or the name of additional information.

When you enter **trace all**, the SDF II trace facility produces additional information, which is identified by an asterisk.

`>` and `<` Indicates whether the trace entry is produced before (`>`) or after (`<`) the service is performed.

This area is blank for program entry (`DGIYAPRC`) or exit (`DGIYARTN`).

`rrrr` The return code (decimal) in register 15.

A zero (`0`) in this position means that the service or function completed normally.

`parameters` A string of parameters that may include the address of the parameter list and the processing time taken.

Abend codes

After an abnormal end, SDF II normally displays panel `DGIYE80`. This panel contains:

- Abend code
- Reason code
- CSECT name

- Offset and ISPF error message number
- Short message
- Long message

If panel DGIYE80 is not displayed, the abend was issued by ISPF or by the operating system.

Appendix D. Using SDF II with ISPF SCLM

This appendix shows a way to integrate SDF II into the application development environment using ISPF SCLM.

The purpose of this appendix is to:

- Help new ISPF SCLM users concerned with SDF II getting started
- Give a detailed description of all customization steps necessary for ISPF SCLM and SDF II
- Provide tips to prevent new users from running into problems

Introduction to ISPF SCLM

The product name, ISPF Software Configuration and Library Manager, indicates that this product combines library manager capabilities and techniques with those of software configuration management:

Library Management Controls data objects (such as source code, object code, panels, and documents) in a set of libraries and maintains accounting information about these objects. Examples of accounting information are:

- Creation date
- Change date
- Owner

Accounting information belongs to the attributes of a data object. The actions performed by the library manager part are those that have a direct effect on single data objects, like create, update, copy, or purge.

Software Configuration Provides the ability to control change within a complete application, which might consist of many data objects. This part uses an additional set of attributes assigned to the objects like object type, method (language), and information about included objects. The description of all data objects, their relationship, and their attributes is called the “configuration” of an application.

The environment for **SC+LM** (software configuration + library management) is defined in a project definition, written by the project administrator. The project definition informs ISPF SCLM about the type of objects (see Figure 74 on page 155) to be controlled, the groups and their hierarchy (see Figure 75 on page 156), the languages that can be assigned to data objects, and some ISPF SCLM control information.

An application can be described with the help of the architecture definition language (ARCHDEF). This language is introduced by ISPF SCLM and allows you to describe the relationship between data objects to be controlled in the project environment. Architecture definitions may include references to data objects as well as references to other architecture definitions.

In this way it is possible to construct an *architecture definition hierarchy* (see Figure 73 on page 152). Architecture definitions are stored in ISPF SCLM controlled libraries like other data objects.

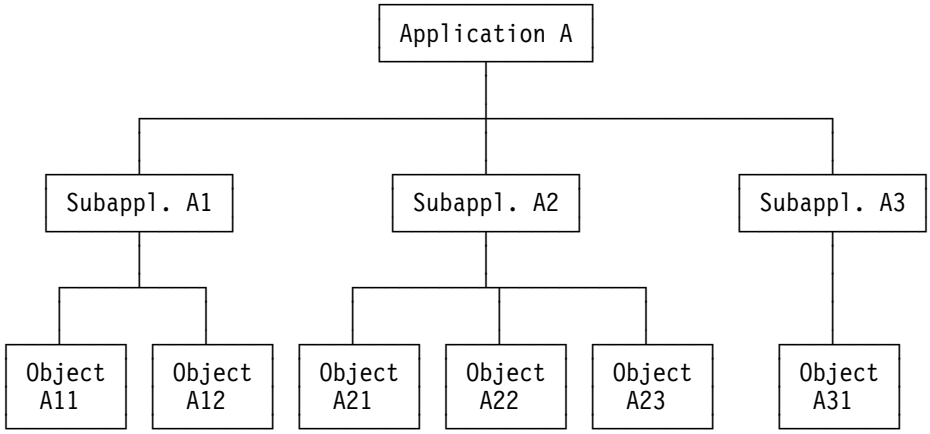


Figure 73. Architecture definition hierarchy

The architecture definition hierarchy must be distinguished from the hierarchy established for the groups defined to the project (*project hierarchy*, see Figure 75 on page 156). The project hierarchy is used to:

- Represent different stages in the development cycle
- Assign the responsibilities for data objects to different groups
- Establish different environments for the groups involved in application development

The main functions provided by ISPF SCLM are Edit, Build, and Promote:

Function	Explanation
Edit	Source objects can be edited using the ISPF editor. For SDF II, you can invoke the SDF II editors with the help of an edit macro. Editing is only possible for groups that belong to the lowest level of the project hierarchy. Data objects that are currently stored for a group on a higher level in the project hierarchy are drawn down and stored in the editors' group when the object is saved. If the object to be edited is currently stored in another branch of the project hierarchy, the object is locked. This prevents two different users, operating at different development levels, from editing the same object at the same time.
Build	Depending on the architecture definition hierarchy level, the Build function performs a translation of a single object as minimum or even a complete application. The Build function will only translate objects that are older than their respective source inputs.
Promote	Moves or copies data objects from the current group to the next higher group defined in the project hierarchy.

SDF II and SCLM in the application development process

ISPF SCLM belongs to the application development platform of AD/Cycle. It is designed to control data objects and their dependencies used in software development. Tools belonging to any phase in the development cycle can be connected to ISPF SCLM in a way that applications can be built and promoted automatically.

SDF II is generally used during the following phases:

- Analysis and Design
- Produce
- Build and Test
- Production and Maintenance

SDF II can be connected to ISPF SCLM as a translator and an editor.

Advantages in using SDF II with SCLM

Some of the advantages of using SDF II with ISPF SCLM are:

- The SDF II SCLM interface allows storage of SDF II objects under the control of a library management system.
- ISPF SCLM allows full integration of SDF II into the application development process. Although SDF II is a complex tool, it can be connected to ISPF SCLM like any other tool.
- ISPF SCLM provides an object-oriented approach for application development. Developers can concentrate on their original tasks, because they do not need to know in detail:
  - How to perform a translation
  - Where to get data objects to be translated
  - Where to put the output
- ISPF SCLM avoids errors that result from using inconsistent data. With the help of architecture definitions you are able to structure the application and perform Build and Promote functions on the application level.
- ISPF SCLM automatically promotes included objects together with their including objects. SDF II uses this capability for include panels and panels referenced in panel groups.

Overview of SDF II implementation for use with SCLM

Implementing SDF II for use with ISPF SCLM requires an SCLM project definition and some adaptation of SDF II user exit routines.

The following list gives a summary of the implementation steps:

1. Allocation of ISPF SCLM project data sets
2. Allocation of SDF II related data sets
3. Identification and allocation of controlled libraries
4. Preparation of SCLM project and language definitions
5. Preparation of the SDF II request file
6. Adaptation of SDF II user exit routines
7. Customization of ISPF SCLM for batch processing

8. Activation of the project definition

As mentioned in “Introduction to ISPF SCLM” on page 151 there are three main functions in ISPF SCLM:

- Edit
- Build
- Promote

The Promote function is managed by ISPF SCLM on its own. SDF II adds ISPF SCLM support for the Edit and Build functions.

Function	Realization with SDF II
Edit	There are two ways to edit SDF II objects controlled by ISPF SCLM: • Via the edit macro using ISPF SCLM dialogs SDF II provides an edit macro to be used as initial macro from ISPF SCLM OPTION 2 EDIT. This macro uses the SDF II invocation interface to invoke the SDF II editor appropriate for the object type. • Via SDF II dialogs The SDF II dialog functions have been enhanced to use ISPF SCLM functions when accessing SDF II objects controlled by ISPF SCLM.
Build	A language definition for each SDF II object type to be controlled by ISPF SCLM must be defined in the ISPF SCLM project definition. Additionally, a Build translator needs to be defined for the object type planned to be used as source for the Build process. The SDF II print and generate functions will be invoked through the SDF II invocation interface.

Getting started with a simple project

This section will lead you through the steps of setting up an SCLM project for use with SDF II objects. These steps match the steps explained in “Defining SCLM-controlled libraries” on page 23. General information about defining projects can be found in chapter 10 “Defining the Project Environment” of *ISPF SCLM Guide and Reference*.

SDF II delivers example language definitions for all supported target systems, which help you create your first project with very little effort. Even if you are not an experienced SCLM administrator, you should be able to set up the project following the steps described here.

For this project, SDF II is the *only* tool defined to ISPF SCLM. Only those objects that are input to or output from SDF II are controlled by ISPF SCLM.

Figure 74 on page 155 shows the ISPF SCLM project environment. The project omits any further translations that could be done to the objects produced by SDF II.

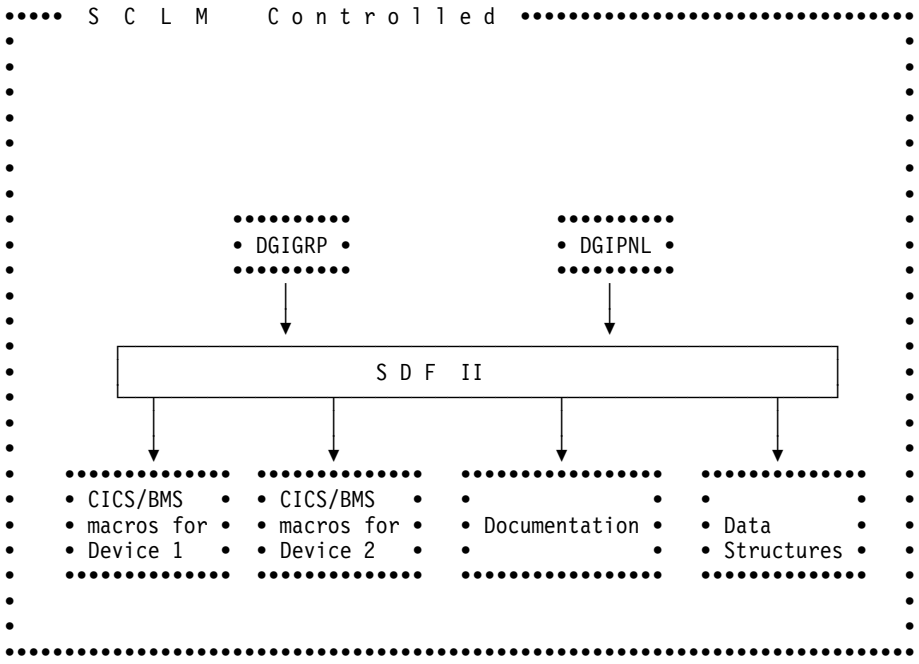


Figure 74. Tool and data flow for sample project

There are at least three reasons for introducing the SDF II ISPF SCLM support:

- Reducing the complexity of the environment helps inexperienced users get started.
- This project emphasizes all those customization steps that are unique to SDF II.
- Users might decide not to restructure old complex applications in a way that they could be completely controlled by ISPF SCLM. Nevertheless they can take advantage of the ISPF SCLM library manager capabilities to store SDF II objects in ISPF SCLM hierarchies and group related objects together. This is *not* the recommended way, but it can be easily done.

Defining the project

The following list gives a summary of the requirements for the project:

Topic	Description
Hierarchy	In Figure 75 on page 156, there are four groups defined in three hierarchy levels. The highest level, in this case the production level, is represented by the group RELEASE, followed by one group for integration testing (INTEGRAT). Two development groups (DEV1 and DEV2) are defined for the lowest level.

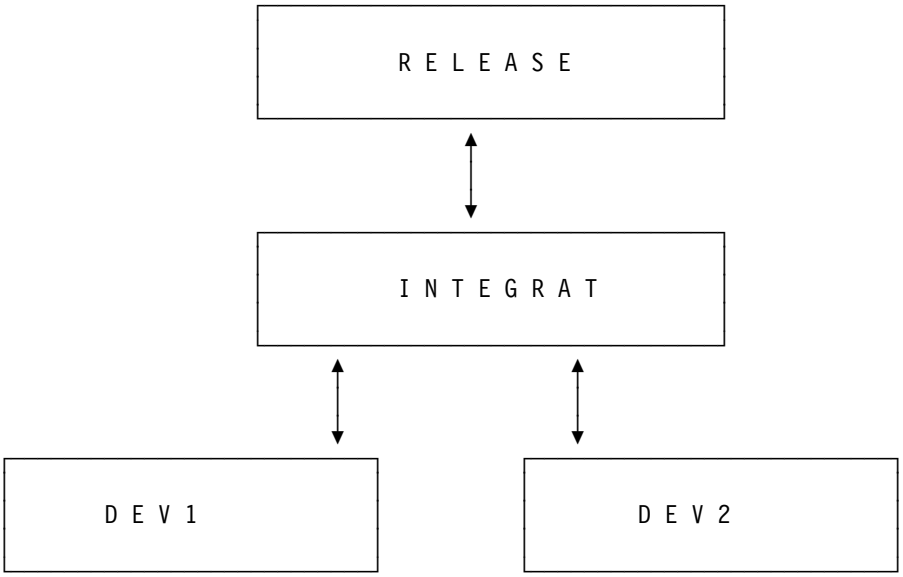


Figure 75. Project hierarchy

Target system	CICS/BMS is used as the target system for SDF II objects.
Object types	The object types DGIGRP (Panel Groups) and DGIPNL (Panels) are defined. No other source objects are to be stored under ISPF SCLM control for this project.
Build process	Object type DGIGRP is used for the build process.
Device types	BMS macros will be generated for two devices (3279-3B and 5550). This assumes **all** objects are defined at least for both devices.
Programming language	SDF II generates data structures for PL/I.
Documentation	Output from the SDF II Print Facility will be saved for documentation purposes.

These requirements should be kept in mind during the whole process of defining the project.

To define the project, you need to take the following steps:

- Allocate SCLM project data sets.
- Allocate SDF II related data sets.
- Identify and allocate controlled libraries.
- Adapt the SCLM project definition.
- Prepare the build request file.
- Tailor user exit routines DGILXLI, DGILXOL, DGILXIO, and DGIIEDT.
- Customize SCLM for batch processing with SDF II.

For convenience, allocate a partitioned data set to contain job control language (JCL), for instance with the name *project*.PROJDEFS.CNTL. This data set will contain the sample JCL provided in this appendix.

Allocating SCLM Project data sets

ISPF SCLM needs the following data sets:

Data set name	Explanation
project.**PROJDEFS.LOAD**	Contains the ISPF SCLM compiled and link-edited project definition. The existence of this data set allows ISPF to determine whether an ISPF SCLM project exists.
project.**PROJDEFS.SOURCE**	Contains project and language definitions written primarily in ISPF SCLM supplied assembler language macros used to define the project to ISPF SCLM.
project.**ACCOUNT.FILE**	A VSAM cluster that contains accounting information about objects stored under control of ISPF SCLM. You can have one or more of these clusters for different groups. Cluster names can be changed. They have to be initialized in addition to being allocated.

Figure 76 on page 158 gives an example of how to allocate project data sets.

```
//SDFDEMOA JOB ...
//*=============================================================*
//*       User action:
//*
//*       Submit this job after you have changed the:
//*       - Jobcard
//*       - Variables at the end of the job
//*       - Volume to contain the accounting data set, if your system
//*         does not use SMS to control such allocations
//*
//PROJALL  EXEC PGM=IEFBR14
//LOAD     DD DSN=SDFDEMO.PROJDEFS.LOAD,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(BLKSIZE=6144,RECFM=U),
//            SPACE=(TRK,(10,5,5)),
//            UNIT=SYSDA
//SOURCE   DD  DSN=SDFDEMO.PROJDEFS.SOURCE,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(10,5,10)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//*=============================================================*
//* Create vsam database for accounting and project database
//*=============================================================*
//ACCTG  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DELETE 'SDFDEMO.ACCOUNT.FILE' CLUSTER
  DEFINE CLUSTER -
    (NAME('SDFDEMO.ACCOUNT.FILE') CYLINDER(4 1) -
    VOLUMES(SDFVOL) -
    KEYS(26 0) IMBED RECORDSIZE(264 32000) SHAREOPTIONS(4,3) -
    SPEED UNIQUE SPANNED ) -
    INDEX(NAME('SDFDEMO.ACCOUNT.FILE.INDEX')) -
    DATA(NAME('SDFDEMO.ACCOUNT.FILE.DATA') CISZ(2024) -
    FREESPACE(50 50))
//STEP2  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//INPUT1 DD  *
  VSAM FILE INITIALIZATION RECORD
/*
//SYSIN    DD *
   REPRO INFILE(INPUT1) OUTDATASET('SDFDEMO.ACCOUNT.FILE')
/*
```

Figure 76. SCLM project data set allocation

Allocating SDF II related data sets

Two data sets will be added by this step that are unique for projects with SDF II. Please note that the data set names used here are only examples:

Data set name	Explanation
project.**PROJDEFS.EXEC**	Will contain user exit routines that are used by SDF II
project.**PROJDEFS.REQUEST**	The request file provides additional parameters to SDF II (see “Preparing the Build Request file” on page 168)

Figure 77 on page 159 shows an example on how to allocate SDF II related data sets.

```
//SDFDEMOB JOB ...
//*=================================================================*
//*
//*       User action:
//*
//*       Submit this job after you have changed the
//*       - jobcard
//*       - variable at the end of the job
//*
//*
//*       variable:
//*          pid       high level qualifier of the project
//*
//*=================================================================*
//GROUP    EXEC PGM=IEFBR14
//EXEC     DD  DSN=&PID..PROJDEFS.EXEC,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(2,2,5)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//REQUEST  DD  DSN=&PID..PROJDEFS.REQUEST,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(1,1,2)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//***********************EXECUTE************************************
//DEM1     EXEC CREATE,PID=SDFDEMO
```

Figure 77. SDF II related data set allocation

Identifying controlled libraries

Controlled libraries are those that contain the real project data, that is, the objects to be controlled by ISPF SCLM. Please refer back to the requirements list in section “Defining the project” on page 155 to identify the controlled libraries:

Library type	Explanation
ARCHDEF	Contains members that describe an application using the ISPF SCLM owned architecture definition language
DGIPNL	SDF II object type panel
DGIGRP	SDF II object type panel group
GRPFMT1	CICS/BMS macros for the first device
GRPFMT2	CICS/BMS macros for the second device
COPY	PL/I data structures
LISTGRP	Listing for panel groups

Note: The library type of a library must be the same as the SDF II object types it contains.

The syntax to construct data set names for ISPF SCLM 3.3 controlled libraries is:

- First qualifier = PROJECT

- Second qualifier = GROUP (see Figure 75 on page 156)
- Third qualifier = TYPE

Figure 78 shows an example for a job that allocates ISPF SCLM controlled libraries.

```
//SDFDEMOC JOB ...
//*=============================================================*
//*
//*       USER ACTION:
//*
//*       Submit this job after you have changed the
//*       - Jobcard
//*       - Variables at the end of the job
//*
//*
//*       Variables:
//*          pid       high level qualifier of the project
//*          group     groups to be used within the project
//*          dir       directory allocation for partitioned data sets
//*
//*=============================================================*
//* Create data sets for group &group in jcl-in-stream procedure
//*=============================================================*
```

Figure 78 (Part 1 of 2). Job control to allocate SCLM controlled libraries

```
//CREATE   PROC
//GROUP    EXEC PGM=IEFBR14
//ARCHDEF  DD DSN=&PID..&GROUP..ARCHDEF,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(2,4,5)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//DGIPNL   DD  DSN=&PID..&GROUP..DGIPNL,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(10,10,&DIR.)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//DGIGRP   DD  DSN=&PID..&GROUP..DGIGRP,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(10,10,&DIR.)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//GRPFMT1  DD  DSN=&PID..&GROUP..GRPFMT1,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(10,10,&DIR.)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//GRPFMT2  DD  DSN=&PID..&GROUP..GRPFMT2,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(10,10,&DIR.)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//COPY     DD  DSN=&PID..&GROUP..COPY,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(10,10,&DIR.)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//LISTGRP  DD  DSN=&PID..&GROUP..LISTGRP,
//           DISP=(NEW,CATLG),UNIT=SYSDA,
//           SPACE=(TRK,(10,10,&DIR.)),
//           DCB=(RECFM=FBA,LRECL=121,BLKSIZE=2420)
//         PEND
//*****************************************************************
//*
//*  In this case we want to allocate data sets for the groups
//*  DEV1, DEV2, INTEGRAT and RELEASE. That means planned are two
//*  development groups, an integration and a production group.
//*
//***********************EXECUTE***********************************
//DEV1     EXEC  CREATE,PID=SDFDEMO,GROUP=DEV1,DIR=3
//DEV2     EXEC  CREATE,PID=SDFDEMO,GROUP=DEV2,DIR=3
//INTEGRAT EXEC  CREATE,PID=SDFDEMO,GROUP=INTEGRAT,DIR=3
//RELEASE  EXEC  CREATE,PID=SDFDEMO,GROUP=RELEASE,DIR=3
```

Figure 78 (Part 2 of 2). Job control to allocate SCLM controlled libraries

Adapting the SCLM project definition

The project definition is the heart of the project and normally is written by the project administrator. A sample project definition is provided in Figure 79 on page 162.

First, note that the project definition consists of nothing but assembler macros, most of which are ISPF SCLM-supplied. Therefore, the rules for writing assembler programs apply. Remember that continuation lines start in column 16.

Some of the most important macros are listed below to provide an overview. See *ISPF SCLM Guide and Reference* for reference information.

Macro name	Explanation
FLMABEG	Defines the name of the project definition.

FLMTYPE Defines the object type name. For ISPF SCLM 3.3 each type must be associated with each group. This restriction has been lifted for ISPF SCLM 3.4.

FLMGROUP Defines the groups in the hierarchy. The hierarchy path is defined by using the PROMOTE parameter.

FLMCNTRL Specifies control options that are valid for the whole project. The ACCT parameter, for example, makes the account file known to ISPF SCLM.

COPY Includes further definitions into the project. The examples in this book use COPY to include language definitions. "Language definition" is an ISPF SCLM term and means the definition of a set of translators to be invoked by ISPF SCLM. Different translators need to be invoked depending on the ISPF SCLM service.

FLMAEND Marks the end of a project definition.

The data set "*projid*.PROJDEFS.SOURCE" should be used to store the source code of project and language definitions.

```
*=======================================================================*
*           USER ACTION:
*
*        - Understand the content and read sclm guide and ref for
*          information about the macros
*        - Change sdfdemo into your project id
*
         TITLE '*** LIBRARY DEFINITION FOR SDFDEMO       ***'
*
SDFDEMO  FLMABEG
*
*        ************************************************************
*        *  DEFINE THE TYPES                                        *
*        ************************************************************
ARCHDEF  FLMTYPE       SCLM ARCHITECTURE DEFINITIONS
DGIGRP   FLMTYPE       SDF II OBJECTS OF TYPE PANEL GROUP
DGIPNL   FLMTYPE       SDF II OBJECTS OF TYPE PANEL
LISTGRP  FLMTYPE       LISTINGS OF THE PANEL GROUP
COPY     FLMTYPE       GENERATED DSECTS FOR LANGUAGE PLI
GRPFMT1  FLMTYPE       GENERATED BMS MACROS FOR MAPSETS FOR DEVICE 1
GRPFMT2  FLMTYPE       GENERATED BMS MACROS FOR MAPSETS FOR DEVICE 2
*
*        ************************************************************
*        *  DEFINE THE GROUPS                                       *
*        ************************************************************
DEV1     FLMGROUP AC=(P),KEY=Y,PROMOTE=INTEGRAT
DEV2     FLMGROUP AC=(P),KEY=Y,PROMOTE=INTEGRAT
INTEGRAT FLMGROUP AC=(P),KEY=Y,PROMOTE=RELEASE
RELEASE  FLMGROUP AC=(P),KEY=Y
```

Figure 79 (Part 1 of 2). Project definition

```
*
*********************************************************************
*                PROJECT CONTROLS
*********************************************************************
*
         FLMCNTRL ACCT=SDFDEMO.ACCOUNT.FILE,               C
               OPTOVER=N
*
*********************************************************************
*                LANGUAGE DEFINITION TABLES
*********************************************************************
*
*
         COPY  FLM@ARCD           -- ARCHITECTURE DEFINITION    --
         COPY  DGIILAN1           -- SDF II BMS PANEL GROUP      --
*
*********************************************************************
*
         FLMAEND
```

Figure 79 (Part 2 of 2). Project definition

SDF II language definition: You may have found the project definition rather simple. The included language definition DGIILAN1, which is provided by SDF II, needs a little more explanation. Again, the macros used in the language definition will be explained first.

Macro name	Explanation
FLMLANGL	Defines a language to SCLM.
FLMTRNSL	Defines a translator. Five functions may be identified: Parse, Verify, Build, Copy, and Purge. For SDF II, only the Build translator needs to be defined.
FLMALLOC	Defines allocations similar to DD statements in job control language.

The language definition as it is displayed in Figure 80 on page 164 is taken from the SDF II shipped version. The only change is for the request data set parameter (parameters will be explained later). The default data set that contains these samples is called “SDF2.V1R4M0.SDGISAM.” For CICS/BMS the language example is in member DGIILAN1. Copy this member to data set “*projid*.PROJDEFS.SOURCE” .

Notes:

1. The allocation of the list data sets depends on the kind of output you want SDF II to generate.
   - Use ddname DGIPRINT if you want to print for standard printer.
   - Use ddname DGIDCF in addition if you want to print DCF output.

     In this case, trace output will be written to ddname DGIPRINT and the documentation will be stored in the data set allocated to DGIDCF.
2. SDF II has been enhanced after delivery of the base code with APAR PN27678 to support printing DCF output under control of ISPF SCLM.

```
*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
*
*      User Action:
*
*      - Change SDFDEMO to your project ID
*
*********************************************************************
*      SCREEN  DEFINITION  FACILITY II  -  RELEASE 4 LEVEL 0
* SCLM language definition example for BMS for the licensed programs:
*  5665-366 (C) COPYRIGHT IBM CORP. 1987, 1994. ALL RIGHTS RESERVED.
*********************************************************************
*
* DGIILAN1 - SCLM language definition example for CICS/BMS
*
*    For CICS/BMS the following languages are possible
*
*         1. DGIGRP - panel groups
*         2. DGIPNL - panels
*         3. DGIPST - partition set
*
*    In this sample the following is assumed:
*      - Objects of type panel(DGIPNL) & panel group (DGIGRP)
*        are handled
*      - BUILD is performed for DGIGRP (not for DGIPNL)
*      - All map sets and maps are defined for 2 devices
*        and will be generated for 2 devices
*      - The DSECTs will be generated for one language (PLI)
*      - The SDF II request file for the build is
*        'projid.PROJDEFS.REQUEST(DGIIGRP1)'
*
*    SDF II uses dynamic include tracking to inform SCLM about
*    related objects therefore no PARSER is required.
*********************************************************************
*===================================================================
*
* LANGUAGE DGIGRP - Panel groups
*
*===================================================================
      FLMLANGL   LANG=DGIGRP,VERSION=V1R4M0
*
```

Figure 80 (Part 1 of 2). Panel group language definition

```
*-------------------------------------------------------------------
* BUILD TRANSLATOR
*-------------------------------------------------------------------
      FLMTRNSL CALLNAM='SDF2 GENERATION ',                          C
               FUNCTN=BUILD,                                        C
               COMPILE=IRXJCL,                                      C
               CALLMETH=LINK,                                       C
               PORDER=1,                                            C
               VERSION=V1R4M0,                                      C
               OPTIONS='DGIIXISP DGIIXBLD @@FLMINC                  C
               ''SDFDEMO.PROJDEFS.REQUEST(DGIIGRP1)'' / @@FLMGRP    C
               @@FLMTYP @@FLMMBR @@FLMPRJ DGIDS1 DGIFMT1 DGIFMT2'
*
*-------------------------------------------------------------------
* Allocate listing data set
*-------------------------------------------------------------------
      FLMALLOC IOTYPE=O,                                            C
               KEYREF=LIST,DDNAME=DGIPRINT,DFLTTYP=LISTGRP,         C
               RECFM=FBA,LRECL=121,RECNUM=500,PRINT=Y
*
*-------------------------------------------------------------------
* Allocate data set for the generated data structure
*-------------------------------------------------------------------
      FLMALLOC IOTYPE=O,                                            C
               KEYREF=OUT1,DFLTTYP=COPY,DDNAME=DGIDS1,              C
               RECFM=FB,LRECL=80,RECNUM=1000
*
*-------------------------------------------------------------------
* Allocate data set for the generated CICS/BMS macro for device 1
*-------------------------------------------------------------------
      FLMALLOC IOTYPE=O,                                            C
               KEYREF=OUT2,DFLTTYP=GRPFMT1,DDNAME=DGIFMT1,          C
               RECFM=FB,LRECL=80,RECNUM=1000
*
*-------------------------------------------------------------------
* Allocate data set for the generated CICS/BMS macros for device 2
*-------------------------------------------------------------------
      FLMALLOC IOTYPE=O,                                            C
               KEYREF=OUT3,DFLTTYP=GRPFMT2,DDNAME=DGIFMT2,          C
               RECFM=FB,LRECL=80,RECNUM=1000
*
*===================================================================
*
* LANGUAGE DGIPNL - Panels
*
*===================================================================
      FLMLANGL   LANG=DGIPNL,VERSION=V1R4M0
*
```

Figure 80 (Part 2 of 2). Panel group language definition

Two languages (DGIGRP and DGIPNL) are defined in this member but only one translator, a Build translator, which is defined for language DGIGRP. Because of this definition you are able to perform a Build from SCLM on SDF II objects of type DGIGRP, which have a language assigned that is called DGIGRP.

SDF II panel groups (DGIGRP) are the input to the Build translator for language DGIGRP. These have includes to SDF II panels (object type DGIPNL, see Figure 81 on page 166). The language definition DGIPNL is used to make these objects known to SCLM.

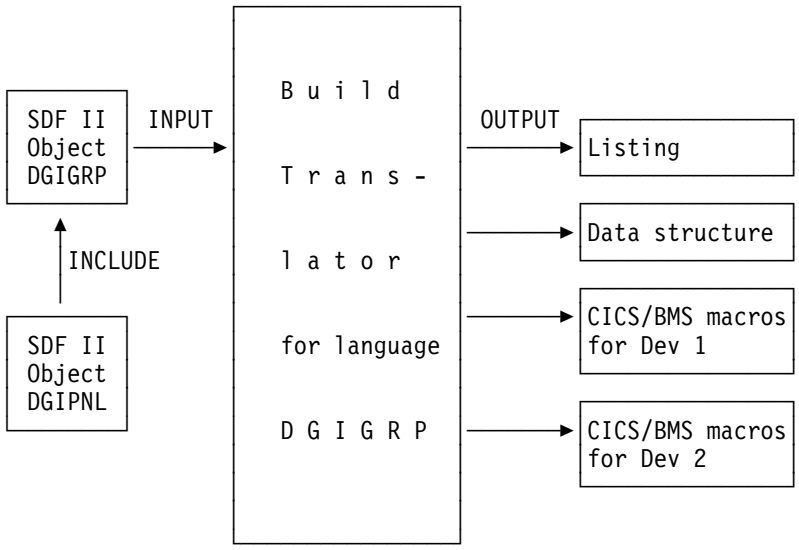


Figure 81. Build translator for DGIGRP

The following paragraphs explain some parameters for the FLMTRNSL macro in detail:

- Figure 82 on page 167 shows the control flow for the Build translator DGIGRP. The Build translator is invoked through the REXX interface **IRXJCL**. The parameters specified under the OPTIONS keyword of the **FLMTRNSL** macro are passed to IRXJCL, which takes the first argument and invokes DGIIXISP implicitly as a REXX EXEC. **DGIIXISP** is a REXX EXEC that invokes the **DGIIXBLD** EXEC through ISPF service `SELECT CMD(...)` to make sure that ISPF services can be used by the subsequent EXECs. Beginning with the next EXEC, DGIIX, the processing might look familiar to you. This is the same interface as used for batch processing.
- **CALLMETH=LINK** is important to keep the ISPF environment.
- **PORDER=1** means the string from the OPTIONS keyword will be passed to the translator as a parameter list.
- *@@FLMINC* is an SCLM variable used to point to included SDF II panels. This variable is used for dynamic include tracking, which is a method to track includes instead of using a parser. Dynamic include tracking is further described in "Building SDF II objects" on page 187.
- *projid*.PROJDEFS.REQUEST(DGIIGRP1) is the request file that is needed for the SDF II invocation interface. This sample request file is also delivered with SDF II and can be found in data set SDF2.V1R4M0.SDGISAM, member `DGIIGRP1`.

  The following parameters must be passed after the slash in the sequence indicated in the example:

  – *@@FLMGRP*, *@@FLMTYP*, *@@FLMMBR*, and *@@FLMPRJ* are SCLM variables that contain group, type, member, and project of the object to be built.

  – DGIDS1, DGIFMT1, and DGIFMT2 are ddnames that tell SDF II where to store its output. The ddnames are used in the request file (see "Preparing the Build Request file" on page 168).

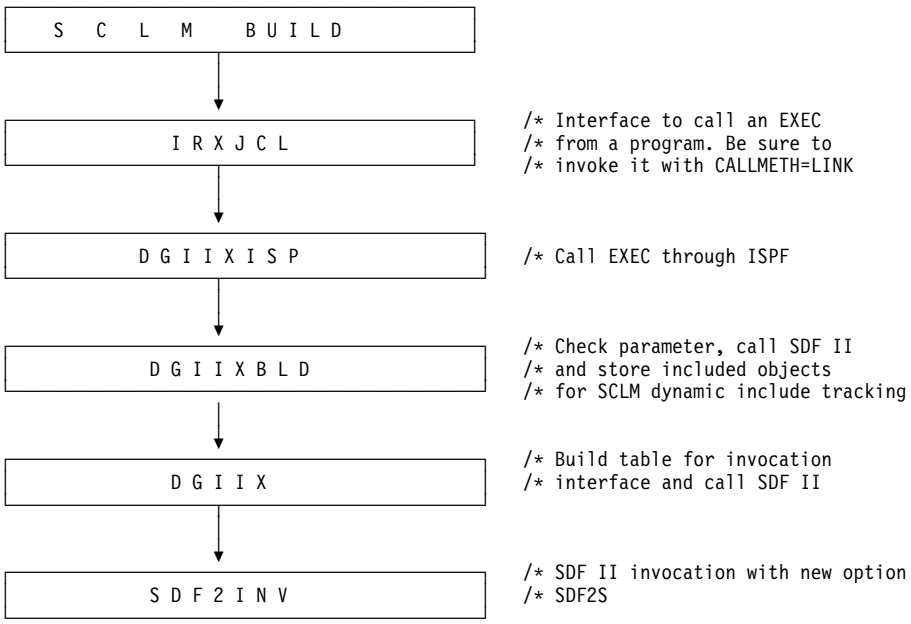


Figure 82. Control flow for Build translator

Activating the project: Finally it is necessary to assemble and link-edit the project definition and store the load module in the "*projid*.PROJDEFS.LOAD(*projid*)" data set. The name for this data set is fixed. The same applies for the member name, which is the primary project definition.

Figure 83 on page 168 shows an example job for this purpose. The SYSLIB concatenation needs the project definition source data set first, followed by the ISPF macro library, and finally the system macro library.

```
//SDFDEMOE JOB ...
//          MSGLEVEL=(1,1),NOTIFY=SDFDEMO
//*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
//*
//*   User Actions:
//*
//* 1. Change SDFDEMO to your userid.
//* 2. Understand the meaning of the SYSLIB concatenation for ASMH
//* 3. Have a look on the SYSIN for ASMH
//*
//CREATE   PROC
//*------------------------------------------------------------ ASSEMBLE
//ASMH     EXEC PGM=IEV90,PARM='OBJECT,NODECK,RENT'
//SYSLIN   DD DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(5,5,0)),
//            DCB=(BLKSIZE=400),DSN=&&LOADSET
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD DUMMY,DCB=BLKSIZE=80
//SYSLIB   DD DSN=&PROJID..PROJDEFS.SOURCE,DISP=SHR
//         DD DSN=ISP.SISPMAC,DISP=SHR
//         DD DSN=SYS1.MACLIB,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(1700,(100,50))
//SYSUT2   DD UNIT=SYSDA,SPACE=(1700,(100,50))
//SYSUT3   DD UNIT=SYSDA,SPACE=(1700,(100,50))
//SYSIN    DD DSN=&PROJID..PROJDEFS.SOURCE(&PROJDEF),DISP=SHR
//*------------------------------------------------------------ LINK
//LKED     EXEC PGM=IEWL,PARM='XREF,LIST,RENT',COND=(0,NE)
//SYSLIN   DD DSN=&&LOADSET,DISP=(OLD,DELETE)
//         DD DDNAME=SYSIN
//SYSLMOD  DD DSN=&PROJID..PROJDEFS.LOAD(&PROJDEF.),DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(1024,(50,20))
//SYSPRINT DD SYSOUT=*
//         PEND
//************************************************************ EXECUTE
//SCLM0    EXEC CREATE,PROJID=SDFDEMO,PROJDEF=SDFDEMO
//*****************************************************************
```

Figure 83. Job control to assemble and linkedit the project definition

Preparing the Build Request file

The SDF II invocation interface is used for interfacing with ISPF SCLM. This is the same interface used to invoke SDF II in batch, see “Step 4: Preparing build request files” on page 27 and Chapter 4, “Interfacing with SDF II” on page 37.

The Build Request file for this project is provided in data set SDF2.V1R4M0.SDGISAM(DGIIGRP1) and shown in Figure 84 on page 169.

The header informs you about the variables used in the request file. If you refer back to Figure 81 on page 166, you can locate these arguments in the OPTIONS list of the FLMTRNSL macro as parameters passed after the backslash.

Note: The request file contains instructions to print a panel group during Build process, but there is no ddname passed to the translator to tell SDF II where to print the object. This is not necessary because SDF II prints to:

- ddname DGIPRINT, if the output is for standard printer
- ddname DGIDCF, if the output is for DCF

```
*********************************************************************
*      SCREEN  DEFINITION  FACILITY II  -  RELEASE 4 LEVEL 0
*    SCLM request file example for BMS for the licensed programs:
*  5665-366 (C) COPYRIGHT IBM CORP. 1987, 1995. ALL RIGHTS RESERVED.
*********************************************************************
** Example request file for an SCLM controlled build of a CICS/BMS
** panel group
**
** This example assumes that the BMS panel group has been defined for
** the devices 3279-3B and 5550. Therefore the generation is called
** twice.
**
** WORD($args,1) -- @@FLMGRP
** WORD($args,2) -- @@FLMTYP
** WORD($args,3) -- @@FLMMBR
** WORD($args,4) -- @@FLMPRJ
** WORD($args,5) -- ddname to be used for DSECT generation
** WORD($args,6) -- ddname to be used for BMS macro generation (3279-3B)
** WORD($args,7) -- ddname to be used for BMS macro generation (5550)
*********************************************************************
ILV00ACC='0'               /* dont access CMS minidisks              */
ILV00PDF='1'               /* use PDF                                */
ILV10L01="WORD"($args,4)   /* library name                           */
ILV10L02=''                /*                                        */
ILV10L03=''                /*                                        */
ILV10L04=''                /*                                        */
ILV10L05=''                /*                                        */
ILV10L06=''                /*                                        */
ILV10L07=''                /*                                        */
ILV10L08=''                /*                                        */
ILV10L09=''                /*                                        */
GRP1    ="WORD"($args,1)   /* lowest SCLM group                      */
IYV50PRT='Y'               /* print messages                         */
IIVREQ='P'                 /* P R I N T                              */
IIVNAM="WORD"($args,3)     /* Object name                            */
IIVLIB='1'                 /* from library                           */
IIVTYP='G'                 /* Type is panel group                    */
IIVPRO='B'                 /* batch                                  */
IUV10FMT='1'               /* output is for standard printer         */
IUV10CNT='1'               /* complete listing requested             */
*
```

Figure 84 (Part 1 of 2). Request file for Build translator

```
IIVREQ='G'                /* G E N E R A T E  3 2 7 9  -  3 B      */
IIVNAM="WORD"($args,3)    /* Object name                           */
IIVTYP='G'                /* Type is panel group                   */
IIVLIB='1'                /* From library                          */
IIVMOD='1'                /* BMS                                   */
IIVPRO='B'                /* batch                                 */
ICV10GDS='Y'              /* Generate DSECT                        */
ICV10DLA='PLI'            /*                For Language           */
ICV10DAL='Y'              /*                Alignment ?            */
ICV10DGR='N'              /*                Graphic ?              */
ICV10DTR='N'              /*                Trigraph ?             */
ICV10DUE=''               /*                User exit              */
ICV10DOL=''               /*                output library         */
ICV10DOD="WORD"($args,5)  /*                DD name                */
ICV10GCB='Y'              /* Generate CICS/BMS macros              */
ICV10CDT='3279-3B'        /*                For device             */
ICV10CUE=''               /*                User exit              */
ICV10COL=''               /*                Output library         */
ICV10COD="WORD"($args,6)  /*                DD name                */
*
IIVREQ='G'                /* G E N E R A T E  5 5 5 0              */
IIVNAM="WORD"($args,3)    /* Object name                           */
IIVTYP='G'                /* Type is panel group                   */
IIVLIB='1'                /* From library                          */
IIVMOD='1'                /* BMS                                   */
IIVPRO='B'                /*                                       */
ICV10GDS='N'              /* No DSECT                              */
ICV10GCB='Y'              /* Generate BMS macros                   */
ICV10CDT='5550'           /*                For Device             */
ICV10COD="WORD"($args,7)  /*                DD name                */
```

Figure 84 (Part 2 of 2). Request file for Build translator

Figure 85 shows how to define the request file for printing DCF output.

```
IIVREQ='P'                /* P R I N T                             */
IIVNAM="WORD"($args,3)    /* Object name                           */
IIVLIB='1'                /* from library                          */
IIVTYP='G'                /* Type is panel group                   */
IIVPRO='B'                /* batch                                 */
IUV10FMT='3'              /* output is for DCF                     */
IUV10CNT='1'              /* complete listing requested            */
IUV10DSN=''               /* print to ddname DGIDCF                */
```

Figure 85. Sample request file for printing DCF

Tailoring user exit routines

DGILXLI: The DGILXLI EXEC checks the type of libraries used with SDF II. Four library types are possible:

- ISPF/PDF libraries
- SCLM-controlled libraries
- Externally controlled libraries
- SCLM-controlled or PDF libraries to be edited via edit macro

The sample exit routine delivered with SDF II can be copied from SDF2.V1R4M0.SDGISAM(DGILXLI) (see Figure 86 on page 172) into the self-defined projid.PROJDEFS.EXEC or into another EXEC library that is concatenated to the ddname SYSEXEC prior to the SDF2.V1R4M0.SDGICMD (which contains a default DGILXLI that simply exits with a return code of 0).

The control flow of DGILXLI is:

- The ILVUELIB variable, which is read from the shared pool at the beginning of DGILXLI, is either the *project name* when SDF II was invoked from SCLM or a *library name* that can be defined in the Specify Libraries panel when using SDF II dialogs.
- Specify the number of libraries used with SDF II in your environment.
- For each library, you have to identify the library ID, the library type, the project name, and the group the user should use. The group set here is only relevant for SDF II edit processing when using SDF II dialogs.

  The example assumes that there is an ISPF SCLM project defined that is named SDFDEMO, and that users have their own library. An installation that has one group used by multiple users might code:

  ```
  if "USERID"() = "XYZ" then grp.1 = "DEV1"
  if "USERID"() = "ABC" then grp.1 = "DEV1"
  if "USERID"() = "BDE" then grp.1 = "DEV2"
   ...
  ```

  If you need to have users working in more than one development group you can achieve this by displaying an ISPF panel that allows a user to select the desired group.
- The DO-loop scans all libraries defined in DGILXLI until the correct one is found.

  Library ID vs. Project Name

  The library ID entered in the Specify Libraries dialog should be the same as the project name. Otherwise you will receive different values, depending on the method used to invoke SDF II—as dialog or by SCLM through the invocation interface.

- Another possibility offered by DGILXLI is to restrict access to PDF libraries. This can be achieved easily by issuing a return code of 8.

```
/* REXX ***************************************************************
*      SCREEN  DEFINITION  FACILITY II  -  RELEASE 4 LEVEL 0
*         Library user exit example for the licensed programs:
*  5665-366 (C) COPYRIGHT IBM CORP. 1987, 1995. ALL RIGHTS RESERVED.
*********************************************************************
*
* DGILXLI - Sample EXEC for library support
*
* This EXEC is called for each SDF II library defined in the
* SPECIFY LIBRARIES dialog.
* Define here the libraries you want to use.
* For a detailed description see "Installation and Customization Guide"
*
* This EXEC has 3 functions:
*
*   1. Check the name only
*   2. Init/Setup the library
*   3. Finis/Close the library
*
* Input: Arg(1) ..... The function.
*                     "C" for Check only
*                     "I" for Init/Setup
*                     "F" for finis/Close
*        ILVUELIB ... The library name
*        ILVUELUD ... any user data related to the library
*                     (only for function "F")
*
* Output: (only for function "I")
*        ILVUEPNM ... Project name (for SCLM controlled libraries only)
*        ILVUEGN1 ... group name (for SCLM controlled libraries only)
*        ILVUELUD ... any user data related to the library (optional)
*
* Return codes: 0 ... OK - it is an SDF II controlled PDF library
*               1 ... OK - it is an SCLM controlled library
*               2 ... OK - it is an external controlled library
*               8 ... Invalid library name
*              32 ... Internal error detected
*
*********************************************************************/
  Parse source env .
/*-------------------------------------------------------------------
* Get the input parms
*-------------------------------------------------------------------*/
  Parse Upper Arg request          /* get the function               */

  ispcmd="VGET ILVUELIB SHARED"    /* get the library name           */
  Address ISPEXEC ispcmd           /*                                */
  If rc¬=0 Then Signal ISPERR      /*                                */
```

Figure 86 (Part 1 of 5). DGILXLI user exit routine

```
/*********************************************************************
* Definition of supported libraries
*
* Add here your libraries
*   name.* ... The library is as entered in the SPECIFY LIBRARIES dialog
*   type.* ... an internal code to indicate the type of library.
*              Is used in the user exits only to select the service
*              routine
*   proj.* ... For SCLM it keeps the high level name
*   grp.*  ... For SCLM it keeps the second level name of the lowest
*              group in the path to be used.
*
* In this sample 2 libraries are defined:
*  1. An SCLM controlled library
*  2. An external controlled library to invoke an SDF II editor via
*     ISPF/PDF edit macro. The library ID must be defined in user exit
*   routine DGIIEDT too.
*********************************************************************/
libs=2                              /* number of libraries            */

name.1="SDFDEMO"                    /* id of library                  */
type.1="SCLM"                       /* type of library                */
proj.1=name.1                       /* SCLM - project name            */
grp.1="USERID"()                    /* SCLM - lowest layer            */


name.2="SDF2PDF"                    /* id of library                  */
type.2="ISREDIT"                    /* type of library                */
/*-------------------------------------------------------------------
* Check the passed library name
*-------------------------------------------------------------------*/
  Do i=libs by -1 to 1 ,            /* scan all defined libraries    */
     Until ilvuelib=name.i          /* id found                       */
    End                             /* end of loop                    */
/*  If you want to restrict the usage of libraries to the ones       */
/*  defined in this EXEC, change the return code of the              */
/*  following Exit command to 8.                                     */
  If i<=0 Then                      /* name not found                 */
    Exit 0                          /* a PDF library                  */

/*-------------------------------------------------------------------
* Select the action routine
*-------------------------------------------------------------------*/
  Select                            /* function                       */
    When type.i="PDF"  Then         /* it is a PDF library            */
      Exit 0                        /* will be handled by SDF II      */
    When type.i="SCLM" Then Signal SCLM /* SCLM controlled library    */
    When type.i="EXT"  Then Signal EXT  /* External controlled        */
    When type.i="ISREDIT" Then Signal ISREDIT /* Via edit macro       */
    Otherwise Exit 32               /* invalid library type           */
    End                             /*                                */
```

Figure 86 (Part 2 of 5). DGILXLI user exit routine

```
SCLM:
/*********************************************************************
*
*                     SCLM controlled libraries
*
* For the CHECK request no additional checks are required
* For the INIT request of SCLM controlled libraries SDF II needs
*   ILVUEPNM - the project name
*   ILVUEGN1 - name of the lowest group to be scanned
*    - for BUILD this is determined by the build layer
*      and is passed by the calling build translator DGICXBLD
*    - for the other functions it is the development layer,
*      normally the user ID of the user
* For the FINIS request no processing is required
*
*********************************************************************/
  Select                            /* function                     */
    When request="C" Then Nop       /* perform a name check         */

    When request="I" Then Do        /* perform init function        */
     If env='CMS' Then Do
      /*-------------------------------------------------------------*/
      /* Set variable ZFLMDCSS in ISRPROF                            */
      /*-------------------------------------------------------------*/
      ispcmd="CONTROL NONDISPL END" /* no display of panel ISR@PRIM  */
      Address ISPEXEC ispcmd        /*                              */
      If rc¬=0 Then Signal ISPERR   /*                              */

      ispcmd="SELECT PANEL(ISR@PRIM) NEWAPPL(ISR) PASSLIB"
      Address ISPEXEC ispcmd        /* write ISRPROF (init ZFLMDCSS) */
      If rc¬=0 & rc¬=4 Then Signal ISPERR /*                        */
      End

      /*-------------------------------------------------------------*/
      /* Set variables ILVUEPNM and ILVUEGN1 for SDF II              */
      /*-------------------------------------------------------------*/
      ilvuepnm=proj.i               /* set project name             */

      ispcmd="VGET GRP1"            /* get group name passed        */
      Address ISPEXEC ispcmd        /*                              */
      If rc¬=0 & rc¬=8 Then Signal ISPERR /*                        */
      If grp1¬='' Then              /* a group name is passed       */
        ilvuegn1=grp1               /* take passed group name       */
      Else                          /* no group name passed         */
        ilvuegn1=grp.i              /* take development layer       */
```

Figure 86 (Part 3 of 5). DGILXLI user exit routine

```
      /*-------------------------------------------------------------*/
      /* Set variable ILVUELUD to keep info for the other user exits */
      /*-------------------------------------------------------------*/
      ilvuelud=type.i ilvuepnm ilvuegn1 /* set user data             */

      ispcmd="VPUT (ILVUEPNM ILVUEGN1 ILVUELUD) SHARED" /*           */
      Address ISPEXEC ispcmd        /*                              */
      If rc¬=0 Then Signal ISPERR   /*                              */
      End                           /* of init function             */

    When request="F" Then Nop       /* termination function         */

    Otherwise Exit 32               /* invalid function             */
    End

  Exit 1                            /* SCLM controlled library      */

EXT:
/*********************************************************************
*
*                   External controlled library
*
* This is a  sample of a simple single level library working similar
* to an SDF II library.
* The objects are stored in partitioned data sets, no locking is
* performed.
*********************************************************************/
  Select                            /* function                     */
    When request="C" Then Nop       /* perform a name check         */

    When request="I" Then Do        /* perform init function        */

      /*-------------------------------------------------------------*/
      /* Set variable ILVUELUD to keep info for the other user exits */
      /*-------------------------------------------------------------*/
      ilvuelud=type.i proj.i grp.i  /* set user data                */
      ispcmd="VPUT ILVUELUD SHARED" /*                              */
      Address ISPEXEC ispcmd        /*                              */
      If rc¬=0 Then Signal ISPERR   /*                              */
      End                           /* of init function             */

    When request="F" Then Nop       /* termination function         */

    Otherwise Exit 32               /* invalid function             */
    End                             /*                              */

  Exit 2                            /* it is an external control'd lib*/
```

Figure 86 (Part 4 of 5). DGILXLI user exit routine

```
ISREDIT:
/*********************************************************************
*
*                   Invoked via edit macro
*
*********************************************************************/
  Select                            /* function                     */
    When request="C" Then Nop       /* perform a name check         */

    When request="I" Then Do        /* perform init function        */

      /*-------------------------------------------------------------*/
      /* Set variable ILVUELUD to keep info for the other user exits */
      /*-------------------------------------------------------------*/
      ilvuelud=type.i               /* set user data                */
      ispcmd="VPUT ILVUELUD SHARED" /*                              */
      Address ISPEXEC ispcmd        /*                              */
      If rc¬=0 Then Signal ISPERR   /*                              */
      End                           /* of init function             */

    When request="F" Then Nop       /* termination function         */

    Otherwise Exit 32               /* invalid function             */
    End                             /*                              */

  Exit 2                            /* it is an external control'd lib*/

ISPERR:
/*********************************************************************
* Handle errors returned from ISPF
*********************************************************************/
/* set:       env typc en et em namc addr                            */
  Parse Source $e  $c  $n $t $m  $i   $d
  Say $n": unexpected return code detected in line" FORMAT(sigl-1)
  Say "FORMAT"(sigl-1)": Address ISPEXEC '"ispcmd"'"
  Say $n": the service returned with RC="rc
  Say "'"zerrlm"'"
/*--------------------------------------------------------------------
* Return the ISPF message
*-------------------------------------------------------------------*/
  ilvuemsg='DGILM308'
  ilvuesm=zerrsm
  ilvuelm=zerrlm
  ilvuehm=zerrhm
  Address ISPEXEC "VPUT (ILVUEMSG ILVUESM ILVUELM ILVUEHM) SHARED"
Exit 32
```

Figure 86 (Part 5 of 5). DGILXLI user exit routine

DGILXOL: The DGILXOL exit routine is called from the SDF II List Objects dialog and from the SDF II Print Dialog (see Figure 87 on page 177) to ask ISPF SCLM whether SDF II objects exist in the project hierarchy path.

The EXEC needs to be copied from SDF2.V1R4M0.SDGISAM(DGILXOL) to a library allocated to SYSEXEC (that is, *projid*.PROJDEFS.EXEC).

The only change that might be done to the DGILXOL exit routine is to remove the ALLOC and FREE commands and put them into the logon procedure to enhance performance.

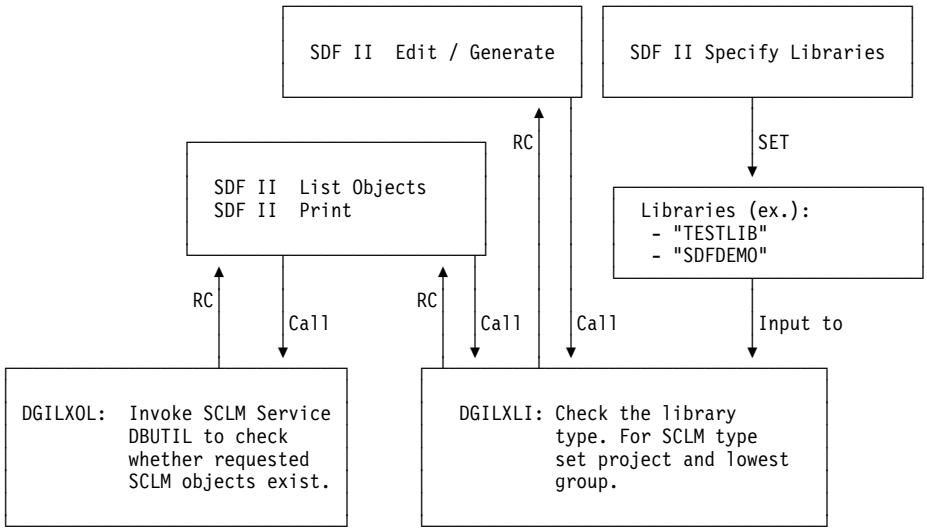


Figure 87. Interrelation between SDF II dialogs and user exit routines

DGIIEDT: The DGIIEDT EXEC is an edit macro provided by SDF II that invokes the SDF II editor from the ISPF editor and can be used from ISPF SCLM Option 2 as an initial macro.

You need to copy the edit macro from SDF2.V1R4M0.SDGISAM to a library concatenated to SYSEXEC. Figure 88 on page 178 shows those parts of DGIIEDT that need customization. The variables ILV10L0x are normally set in the Specify Libraries dialog and represent the libraries to be edited from SDF II. In this example, three possibilities are shown:

Library ID	Objective
SDFDEMO	Defines the library ID SDFDEMO to the exit. This library ID is checked in DGILXLI and detected as ISPF SCLM-controlled library. This setup is necessary for SDF II to find included objects of the member currently selected for editing.
project.group	Defines a normal PDF library to DGIIEDT.
SDF2PDF	This is an unconditional definition of the currently selected member. DGILXLI has to check this library ID and assign the library type ISREDIT to signal to SDF II that an object is edited through an edit macro.

```
/*********************************************************************
* ASSIGNMENT SECTION
* Adapt these parameters to suit your local requirements
*********************************************************************/
/*-------------------------------------------------------------------
* Specify the library to be used for included objects and the target
* system for new objects.
* Device and skeleton object are optional.
*-------------------------------------------------------------------*/
  If minidisk="YES" Then ILV10ACC="1"   /* access CMS minidisks      */
  Else Do                               /* ISPF     library          */
    Parse Value ds With project "." group "."
    Select
      When project="SDFDEMO" Then
        ILV10L01=project
      Otherwise
        ILV10L01="'"||project||"."||group||"'"
    End
  End

  ILV10L02="SDF2PDF"                    /* get object via ISPF       */
  IIVLIB="2"                            /*   editor (edit macro)     */

  IIVDEV=" "                            /* device from the profile   */
  IIVONAM=" "                           /* second (skeleton) object  */
  IIVOLIB=" "                           /*   name and library        */
                                        /*   (profile values used)   */

  DGIRVTS="*"                           /* target system:            */
                                        /*   C  =  CICS/BMS          */
                                        /*   M  =  IMS/MFS           */
                                        /*   I  =  ISPF              */
                                        /*   G  =  GDDM-IMD          */
                                        /*   X  =  CSP/AD            */
                                        /*   *  =  ALL               */
```

Figure 88. Adaptable parts of DGIIEDT

DGILXIO: The original intent of the DGILXIO user exit routine was to process I/O requests for externally controlled libraries. It is also used to copy a data object from the ISPF editor to a temporary data set for later use by an SDF II editor.

The DGILXIO exit is in SDF2.V1R4M0.SDGISAM. All you have to do is to copy the member to a data set that is concatenated to the SYSEXEC ddname (*projid*.PROJDEFS.EXEC).

DGILXOL: The DGILXOL exit is in SDF2.V1R4M0.SDGISAM. All you have to do is to copy the member to a data set that is concatenated to the SYSEXEC ddname (*projid*.PROJDEFS.EXEC).

Customizing ISPF SCLM for batch processing

ISPF SCLM functions like Build, Promote, and Migrate can be performed in batch as well as in a foreground session by using either the command EXECUTE (EX) to start a process in foreground or SUBMIT (SUB) to submit a batch job for this purpose.

If you decide to use ISPF SCLM processes in batch, you need to adapt a skeleton provided by ISPF SCLM to your environment and add some changes for use with SDF II.

The example skeleton is in data set ISR.V4RxM0.ISRSLIB(FLMLIBS) . Figure 89 shows a customized version of this skeleton for use with ISPF V3R3 and SDF II.

```
//*
//*******************************************************************
//* STEPLIB LIBRARIES
//*******************************************************************
//*
//STEPLIB    DD DSN=ISP.V4R1M0.ISPLOAD,DISP=SHR
//           DD DSN=ISP.V4R1M0.ISPLPA,DISP=SHR
//*
//*SYSMDUMP   DD DSN=HLQ.SDF0C4,DISP=MOD
//*
//*******************************************************************
//* ISPF/PDF LIBRARIES
//*******************************************************************
//*
//ISPMLIB    DD DSN=ISP.V4R1M0.ISPMENU,DISP=SHR  ISPF MESSAGES
//*
//ISPSLIB    DD DSN=ISP.V4R1M0.ISPSLIB,DISP=SHR  ISPF SKELS
//*
//ISPPLIB    DD DSN=ISP.V4R1M0.ISPPENU,DISP=SHR  ISPF PANELS
//*
//ISPTLIB    DD DSN=ISP.V4R1M0.ISPTENU,DISP=SHR  ISPF TABLES
//*
//ISPLLIB    DD DSN=ISP.V4R1M0.ISPLOAD,DISP=SHR
//           DD DSN=ISP.V4R1M0.ISPLPA,DISP=SHR
```

Figure 89 (Part 1 of 2). Customized skeleton FLMLIBS

```
//*
//ISPTABL    DD UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB),
//            DSN=&TABLESP                TEMPORARY TABLE LIBRARY
//*
//ISPCTL1  DD DISP=(NEW,PASS),SPACE=(CYL,(10,10)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PS,RECFM=FB),
//            DSN=&ZUSER..BACK.SPFTEMP1.CTL
//*
//ISPPROF  DD UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB),
//            DSN=&TABLESP                TEMPORARY TABLE LIBRARY
//SYSEXEC  DD DSN=SDFDEMO.PROJDEFS.EXEC,DISP=SHR
//         DD DSN=SDF2.V1R4M0.SDGICMD,DISP=SHR
//*
//ISPLOG   DD SYSOUT=*,
//            DCB=(LRECL=120,BLKSIZE=2400,DSORG=PS,RECFM=FB)
//*
//*-----------------------------------------------------------------
//* TEMPORARY CLIST CONTAINING COMMAND TO BE EXECUTED
//*-----------------------------------------------------------------
//SYSPROC DD DSN=&&&&CLIST&STEP,DISP=(OLD,DELETE)
//*
```

Figure 89 (Part 2 of 2). Customized skeleton FLMLIBS

It is important to pre-allocate ISPPROF, ISPTABL, and ISPCTL1 to data sets that are different from those used in the foreground session, otherwise the batch job will fail.

The SDF II EXECs need to be allocated to SYSEXEC.

Using the project

After having successfully defined the project you enter ISPF SCLM dialogs to begin using the project. If you enter option 10 from the ISPF primary option menu, the SCLM Main Menu will be displayed:

```
FLMDMN                          SCLM Main Menu
Option ===>

Enter one of the following options:

   1 View        ISPF View or Browse data
   2 Edit        Create or change source data in SCLM databases
   3 Utilities   Perform SCLM database utility/reporting functions
   4 Build       Construct SCLM-controlled components
   5 Promote     Move components into SCLM hierarchy
   6 Command     Enter TSO or SCLM Commands
   X Exit        Terminate SCLM


SCLM Project Control Information:
   Project . . . . SDFDEMO   (Project high-level qualifier)
   Alternate . . .           (Project definition: defaults to project)
   Group . . . . . DEV1      (Defaults to TSO prefix)
```

Figure 90. ISPF SCLM Main Menu

Enter your project ID and the group you would like to use into the appropriate fields and try to enter the edit dialog to test whether the project definition is correctly applied.

If this fails, it is most probable that something went wrong when assembling or link-editing the project definition. In this case, check the message return code using *ISPF SCLM Guide and Reference*.

The following sections take you through the first steps with SCLM. You will see how to:

- Migrate existing SDF II objects under SCLM control
- Edit SDF II objects using SDF II and SCLM dialogs
- Write architecture definitions for this project
- Build and promote SDF II objects

Migrating SDF II objects under SCLM control

As starting point for the migration, assume that there are existing SDF II objects of type DGIPNL and DGIGRP stored in partitioned data sets that are not controlled by ISPF SCLM.

Copy these members into the appropriate libraries of a *development* group. SCLM allows migration of objects only to groups that are on the lowest level of the hier-

archy (Figure 75 on page 156). Then enter the SCLM Migration Utility (see Figure 91 on page 181), which is done by choosing ISPF SCLM option 3.3.

```
FLMUM#P              SCLM Migration Utility - Entry Panel
Command ===>

Selection criteria:
  Project  . : SDFDEMO
  Group  . . . DEV1
  Type . . . . DGIpnl
  Member . . . *           (Pattern may be used)

Member information:
  Authorization code  . .                Mode  . . . 1  1. Conditional
  Change code . . . . . .                               2. Unconditional
  Language  . . . . . . . DGIpnl                        3. Forced
Output control:
               Ex Sub                    Process . . 1  1. Execute
  Messages . . 3  3    1. Terminal                      2. Submit
  Listings . . 3  3    2. Printer
                       3. Dataset        Printer . .
                       4. None           Volume  . .
```

Figure 91. Migration Utility - Entry Panel

The migration utility writes the first accounting record for the objects already stored in the library and assigns a language to each object. The objects to be migrated for this project are DGIPNL (with language DGIPNL) and DGIGRP (with language DGIGRP). The resulting messages are listed in Figure 92.

```
 BROWSE -- SDFDEMO.MIGRATE.MSGS14 ------------------- LINE 00000000 COL 001 080
 COMMAND ===>                                           SCROLL ===> CSR
******************************** TOP OF DATA  ********************************
FLM32101 - MIGRATION UTILITY INITIATED - 17:52:47 ON 92/07/23
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER DFH$PGA AT 17:52:52, CODE: 0
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER DFH$PGB AT 17:52:54, CODE: 0
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER DFH$PGC AT 17:52:57, CODE: 0
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER DFH$PGD AT 17:53:00, CODE: 0
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER DFH$PGK AT 17:53:02, CODE: 0
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER DFH$PGL AT 17:53:05, CODE: 0
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER FINAL AT 17:53:07, CODE: 0
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER FOOTING AT 17:53:10, CODE: 0
FLM40501 - NO TRANSLATOR INVOKED FOR LANGUAGE: DGIPNL
FLM87107 - SAVE SUCCEEDED FOR MEMBER HEADING AT 17:53:12, CODE: 0
FLM32401 - MIGRATION UTILITY COMPLETED - 17:53:13 ON 92/07/23
******************************** BOTTOM OF DATA ********************************
```

Figure 92. Migration utility messages

Work on objects using SDF II dialogs

If you want to use SDF II dialogs to edit the objects, you need to define the libraries to SDF II as you do when using normal ISPF libraries. Enter option 8 “Specify Libraries” from the Select an SDF II Function panel and enter the libraries you want to use (see Figure 93).

Remember the library names entered here are interpreted by the DGILXLI exit in SDF II. Use the project name as library name to inform SDF II about the library hierarchy defined in the project.

```
DGILE10                     SPECIFY LIBRARIES
===>

  LIBRARY FOR RELATED OBJECTS . . 1    1 - Scan all libraries
                                       2 - Start with library of primary object
                                       3 - Same library as primary object

  ID  LIBRARY NAME ---------------------------------      PASSWORD

  1   TESTLIB_____________________________________
  2   SDF2SCLM____________________________________
  3   SDFDEMO_____________________________________
  4   P49875______________________________________
  5   _____________________________________________
  6   _____________________________________________
  7   _____________________________________________
  8   _____________________________________________
  9   'SDF2.TESTLIB' _____________________________

Optionally, specify AUTOSAVE library

      _____________________________________________
```

Figure 93. Specify Libraries panel

Having defined the project to SDF II, you can edit objects using the SDF II editor dialogs. An example using the object list is shown in Figure 94 on page 183. For any object type you select, SDF II asks ISPF SCLM via the DBUTIL service if objects exist in the project hierarchy.

```
DGIOE00                  SPECIFY SEARCH ARGUMENT
===>

  OBJECT NAME . . . *

  OBJECT TYPE . . . PG          * - All objects
                                P - PANEL         A - AID TABLE
                                G - PANEL GROUP   O - OPERATOR CONTROL TABLE
                                S - PARTITION SET

  LIBRARY . . . . . 3           * - All libraries

  TARGET SYSTEM . . *           * - All target systems
                                B - CICS/BMS      G - GDDM
                                M - MFS           X - CSP
                                I - ISPF          A - ALL

  SORT SEQUENCE . . LN          N - Object name   D - Description
                                L - Library       M - Modification date
                                T - Object type   G - Generation name
```

Figure 94. SDF II Specify Search Argument panel

You may want to refer back to Figure 87 on page 177. Especially if you are experiencing problems like the SDF II message `NO OBJECT FOUND`, and if you do *not* see DBUTIL messages in line mode as listed in Figure 95. These indications normally arise if the checking for the project against the libraries entered on the Specify Libraries panel was not successful.

```
FLM87115 - DBUTIL SUCCEEDED                     AT 09:28:21, CODE:  0
FLM87115 - DBUTIL SUCCEEDED                     AT 09:28:29, CODE:  0
***
```

Figure 95. Messages from ISPF SCLM DBUTIL service

The objects found in the project hierarchy are displayed on the List Objects panel (see Figure 96 on page 184). In contrast to using normal ISPF libraries, the Delete and Rename commands are not supported here. Trying to delete an object stored in SCLM-controlled libraries leads to the message `User Exit Error` and when pressing help you are informed that `User exit DGILXOD returned with RC=16`. You might add the possibility to delete members from the object list by adding appropriate calls to user exit DGILXOD.

To rename objects in ISPF SCLM, three steps (copy, migrate, and delete) are required, because ISPF SCLM does not provide a rename service.

Generation of objects is allowed from here, but this can lead to errors. Users cannot be prevented from generating into SCLM controlled libraries, which leads to inconsistent accounting information in SCLM.

```
DGIOE10                        LIST OBJECTS
===>                                                  Scroll ===> PAGE

SEARCH ARGUMENT: NAME = *        TYPES = PG    LIBRARY = 3  TARG.SYS.= *
SORT SEQUENCE = LN
OBJECT LIST . . . . . . . . . . . . . . . . . . . . COLUMNS 1-7 OF 7, ROW 1 OF 15
    NAME --- LIB TYP OPERANDS   SYST DESCRIPTION -------------------- LAST MOD
''' DFH$PGA  3   P              CICS BMS: MAPSETA  DFH$AGA             92/07/16
''' DFH$PGB  3   P              CICS BMS: MAPSETB  DFH$AGB             92/07/16
''' DFH$PGC  3   P              CICS BMS: MAPSETC  DFH$AGC             92/07/16
''' DFH$PGD  3   P              CICS BMS: MAPSETD  DFH$AGD             92/07/16
''' DFH$PGK  3   P              CICS BMS: MAPSETK  DFH$AGK             92/07/15
''' DFH$PGL  3   P              CICS BMS: MAPSETL  DFH$AGL             92/07/15
''' FINAL    3   P              CICS BMS: MAPSETD  FINAL               92/07/16
''' FOOTING  3   P              CICS BMS: MAPSETD  FOOTING             92/07/16
''' HEADING  3   P              CICS BMS: MAPSETD  HEADING             92/07/16
''' MAPSETA  3   G              CICS BMS: MAPSETA                      92/07/16
''' MAPSETB  3   G              CICS BMS: MAPSETB                      92/07/16
''' MAPSETC  3   G              CICS BMS: MAPSETC                      92/07/16
''' MAPSETD  3   G              CICS BMS: MAPSETD                      92/07/16
''' MAPSETK  3   G              CICS BMS: MAPSETK                      92/07/15
''' MAPSETL  3   G              CICS BMS: MAPSETL                      92/07/15




'                               <=== WORK HERE ON ANOTHER OBJECT

ne commands: C=Copy      D=Delete  R=Rename  P=Print     CV=Convert
             G=Generate  E=Edit    T=Test    ==Repeat command
```

Figure 96. SDF II List Objects panel

Edit SDF II objects in SCLM

Editing of SDF II objects from ISPF SCLM is possible with the help of the edit macro DGIIEDT. If you have successfully customized the macro, you now can use it from ISPF SCLM Option 2, the SCLM Edit - Entry Panel as shown in Figure 97 on page 185.

```
FLMED#P                    SCLM Edit - Entry Panel
Command ===>

SCLM Library:
   Project . . : SDFDEMO
   Group . . . : DEV1     . . . INTEGRAT . . . RELEASE  . . .
   Type  . . . . DGIPNL
   Member  . . .                  (Blank or pattern for member selection list)

Initial Macro  . . DGIIEDT
Profile Name . . .                (If blank, defaults to data set type)

Enter "/" to select option
/  Confirm Cancel/Move/Replace
   Mixed Mode

Change code . . . . . .
Authorization code  . .           (If blank, the default auth code is used)
Parser Volume . . . . .           (If blank, the default volume is used)
```

Figure 97. Edit SDF II objects from ISPF SCLM

Writing architecture definitions

"An architecture definition describes the order in which an application under SCLM control is to be constructed and integrated. Architecture definitions are created and updated by the developers and describe the architecture of an application. They provide specifications to the Build function for data generation, and they provide specifications to the Promote function for the movement of data from one group to another." (*ISPF SCLM Guide and Reference*)

Note: Ensure that you have sufficient information about architecture definitions before continuing.

If SDF II objects are the only editable (input) objects defined to ISPF SCLM, you can use generic and high-level architecture members in your architecture definition.

Concerning this project, the advantage you can take from using architecture definitions is to group several source objects of type DGIGRP together.

The examples used here to demonstrate ISPF SCLM functions are the PL/I samples taken from *CICS/MVS: Application Programmer's Reference*. You find them in the data set `CICSvrm.SAMPLIB`. The following members are used for this sample project:

Type	Names
BMS Source	DFH$PGA - DFH$PGD, DFHPGK, DFHPGL
DGIPNL	DFH$PGA - DFH$PGD, DFHPGK, DFHPGL
DGIGRP	MAPSETA - MAPSETD, MAPSETK, MAPSETL

To prepare the use of these samples you need to import them into SDF II and assign both device types to the objects. Importing the BMS source into SDF II produces objects of type DGIPNL and DGIGRP automatically.

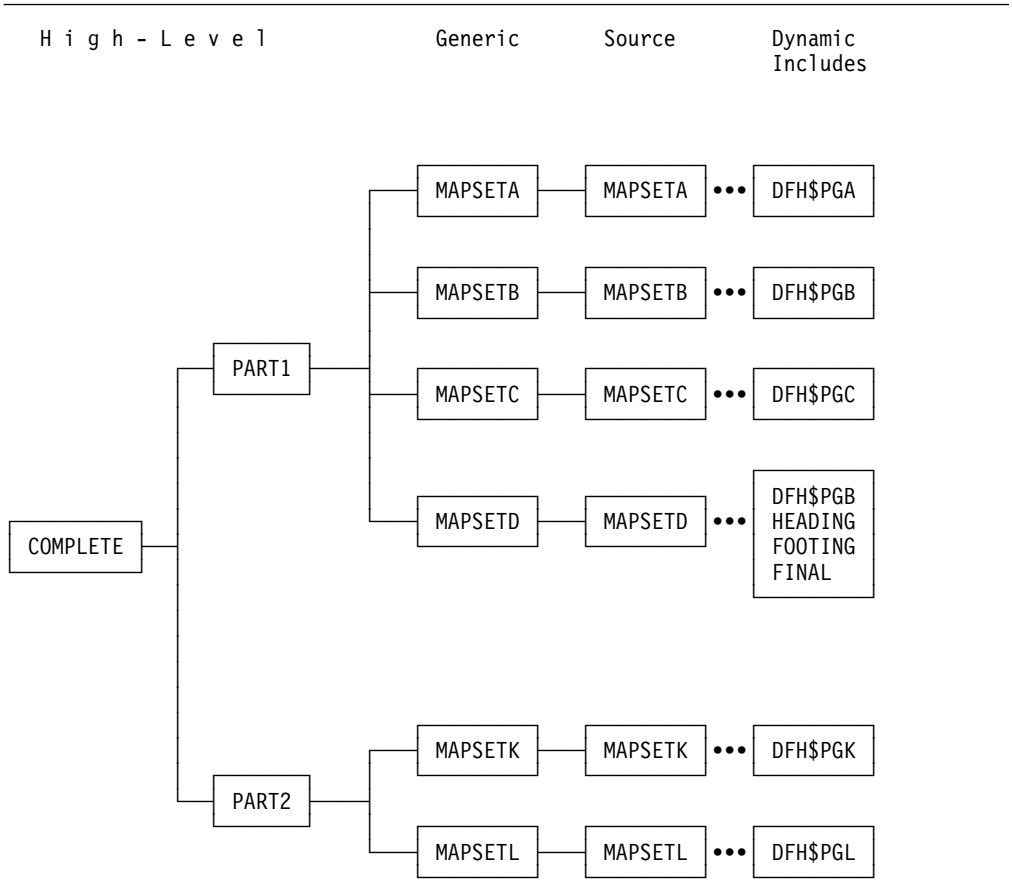


Figure 98. Sample project architecture definition hierarchy

Figure 98 shows the hierarchy of the architecture definition members. COMPLETE is a high-level architecture member and references PART1 and PART2, which are also architecture members of type high-level. PART1 has includes for four generic architecture members (MAPSETA-MAPSETD) and PART2 includes MAPSETK and MAPSETL. Every SDF II object of type DGIGRP needs its own architecture definition member. Figure 99 on page 187 shows an example for each type of architecture definition. The high-level definitions contain exclusively **INCL** statements while the generic definitions depend on the language definition (Figure 80 on page 164). The statements defined in the first column apply to the **KEYREFS**, the next column references the object and the third column determines the object type defined in the project definition, which is the same as the **DFLTTYP**.

The map sets are so-called source objects and internally they have includes for the panels (object type DGIPNL). These objects are not referenced in architecture definition members. Dynamic include tracking will be used to process them correctly with ISPF SCLM.

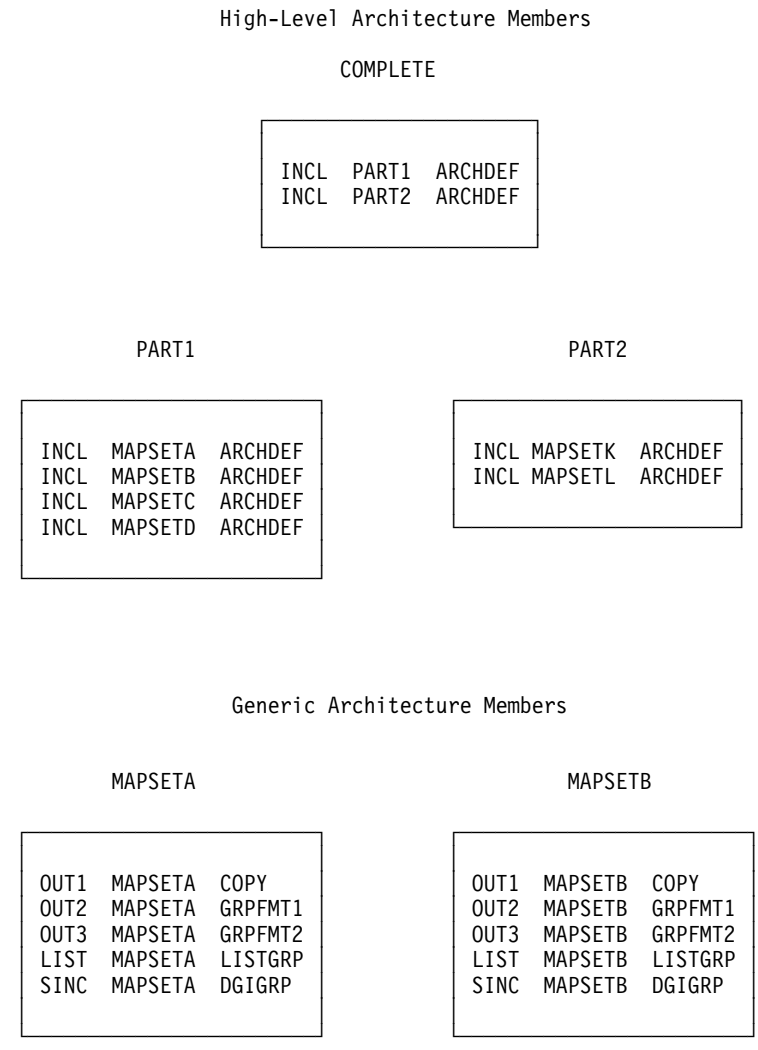


Figure 99. Sample project architecture definitions

Building SDF II objects

Select option 4 on the ISPF SCLM Main Menu to build SDF II panel groups or a set of panel groups.

Figure 100 on page 188 shows the setup for the Build process for building a single object of type DGIGRP.

It is recommended to use batch processing when building a high-level architecture definition, so that you can continue to use your ISPF session for other tasks. ISPF SCLM has to be customized for batch processing as described in “Customizing ISPF SCLM for batch processing” on page 178.

Alternatively, you can use the Build function for any architecture definition level if you replace TYPE and MEMBER on the Build Entry Panel, that is, a Build on

SDFDEMO.DEV1.ARCHDEF(COMPLETE) would cause all objects to be checked whether they are out of date, and rebuilt if necessary.

```
FLMB#P                    SCLM Build - Entry Panel
Command ===>

Build input:
  Project  . : SDFDEMO
  Group  . . . DEV1
  Type . . . . DGIGRP                  Enter "/" to select option
  Member . . . MAPSETA                    Error Listings only

  Mode  . . 1  1. Conditional          Scope . . . 2  1. Limited
               2. Unconditional                       2. Normal
               3. Forced                              3. Subunit
               4. Report                              4. Extended
Output control:
               Ex Sub                  Process . . 1  1. Execute
  Messages . . 3  3    1. Terminal                    2. Submit
  Report . . . 3  3    2. Printer
  Listings . . 3  3    3. Dataset      Printer . . X
                       4. None         Volume  . .
```

Figure 100. SCLM Build - Entry Panel

Promoting SDF II objects

Select option 5 on the ISPF SCLM Main Menu to promote SDF II objects or a set of objects.

Figure 101 shows the SCLM Promote - Entry Panel. In this example a single object is to be promoted. Architecture members can be used instead as described for the Build function.

```
FLMP#P                    SCLM Promote - Entry Panel
Command ===>

Promote input:
  Project . . . : SDFDEMO
  From group  . . DEV1
  Type  . . . . . DGIGRP
  Member  . . . . MAPSETA

  Mode  . . 1  1. Conditional          Scope . . . 1  1. Normal
               2. Unconditional                       2. Subunit
               3. Report                              3. Extended

Output control:
               Ex Sub                  Process . . 1  1. Execute
  Messages . . 3  3    1. Terminal                    2. Submit
  Report . . . 3  3    2. Printer
                       3. Dataset      Printer . . X
                       4. None         Volume  . .
```

Figure 101. SCLM Promote - Entry Panel

Glossary of terms and abbreviations

Glossary terms are defined as they are used in this book. Some definitions have been taken from *American National Standard Dictionary for Information Systems*, in which case they are marked with (A); other definitions are from the *Information Technology Vocabulary*, in which case they are marked with an (I). Definitions without source labels are IBM definitions. If you cannot find the term you are looking for, refer to the index, the online reference index, or to the *IBM Dictionary of Computing,* SC20-1699.

A

Abend. Abnormal end of task.

action bar. In SAA Common User Access architecture, the area at the top of a window that contains choices that give a user access to actions available in that window.

active partition. The partition that contains the cursor. It can be scrolled vertically. While a partition is active, the cursor "wraps around" at the viewport boundaries, and the *ENTER* key (or input key) transmits data from that partition only.

action bar choice. A textual item on an action bar, which provides access to menus that contain choices that can be applied to an object.

adjunct. An optional field in the data structure that is added to a field in the data structure, which contains data to be displayed. It enables the application program to vary a specific presentation attribute (or set of attributes) at run time.

AID. See attention identifier.

AID table. A table that assigns values to actions performed by the user of the application program. An action may be, for example, the pressing of a program function key. The values are used by the application program.

APAR. Authorized program analysis report

application. A collection of software components used to perform specific types of user-oriented work on a computer. Typical examples are payroll applications, airline seat-reservation systems, and stock-control systems.

application attribute. A property of a variable field, such as justification of data in the data structure. Contrast with presentation attribute.

application development (AD). The defining, writing, and testing of a program for a specific solution or application problem.

application element. Any single item in the data structure.

application prototype. A simulation of an application by presenting some or all panels used in the application in a predefined order. See operational prototype and simulative prototype.

area. A rectangular part of a format, whose contents (text or graphics) are provided at run time by the application program. See graphic area and dynamic area.

area attribute. An attribute that affects the properties of an area. It can be, for example, extendable or scrollable.

area mark. A mark used to define an area (see area), such as a graphics area or a dynamic area.

array. A named, ordered collection of variable fields, all of which have identical names and attributes. An array has a specified occurrence number denoting the number of elements in the array. See horizontal array and vertical array.

array index. A number in parentheses that appears next to the name of an array. For example, in the name of the element *a(3)* of the array *a*, *3* is the array index.

assembler (ASM). A computer program that converts assembly language instructions into object code.

attention identifier (AID). A character in a data stream indicating that the user has pressed a key, such as the Enter key, that requests an action by the system.

attribute. See presentation attribute and application attribute. See application attribute, area attribute, background attribute, character attribute, field attribute, inherent attribute, and presentation attribute.

attribute descriptor. A symbol that denotes a set of attributes.

attribute line. A line showing the attribute descriptors assigned to the field.

autosave. An automatic save facility in which the user can define a specific number of alterations after which a temporary save occurs automatically.

autosave library. A library in which the saved objects are stored.

B

background attribute. The attributes associated with background text.

background text. All text on a panel that is not within a constant or variable field.

base name. The name that is used in a based data structure as a pointer variable that identifies the location of the data.

block. In SDF II, a rectangular part of a format that is defined by the position command for such commands as moveblock or delblock.

C

C. A high-level programming language.

character attribute. An attribute that applies to a single character.

CICS/BMS. Customer Information Control System/Basic Mapping Support.

CMS. Conversational monitor system.

COBOL. A high-level programming language, based on English, that is used primarily for business applications.

Common User Access (CUA) architecture. Guidelines for the dialog between a person and a workstation or terminal. One of the three SAA architectural areas. See also Systems Application Architecture solution.

constant field. In SDF II, a field that contains constant text, which has attributes that differ from background attributes. Contrast with variable field.

control table. (1) In IMS/MFS, a user-defined table of operator control functions; a specific control function is invoked when the input device data or data length satisfies a predefined condition. (2) In SDF II, an object that corresponds to an operator control table in IMS/MFS.

controlled library. In SDF II, a library that is controlled by a library system; for example, ISPF/PDF Software Configuration and Library Manager (SCLM). See externally controlled library.

conversational monitor system (CMS). A virtual machine operating system that provides general interactive time sharing, problem solving, and program development capabilities, and operates only under control of the VM/370 VM control program.

conversion. A process by which an object defined for a specific target system is changed so that it becomes an object for another target system. The converted object will retain those properties which are supported by the new target system.

CP. Control program.

Cross System Product (CSP/AD and CSP/AE). A set of licensed programs designed to permit the user to develop and run applications using independently defined maps (display and printer formats), data items (records, working storage, files, and single items), and processes (logic). The Cross System Product set consists of two parts: Cross System Product/Application Development (CSP/AD) and Cross System Product/Application Execution (CSP/AE).

CSP/AD. Cross System Product/Application Development.

CSP/AE. Cross System Product/Application Execution.

CUA. See Common User Access architecture.

CUA attribute. Synonym for CUA panel element attribute.

CUA panel element. The smallest named part of a panel, such as a title, which is based on CUA architecture.

CUA panel element attribute. In SDF II, any attribute associated with a CUA panel element type. Synonymous with CUA attribute.

CUA panel element type. In SDF II, used as a reference to a class of CUA panel elements. Synonymous with CUA type.

CUA type. Synonym for CUA panel element type.

Customer Information Control System (CICS). An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. It includes facilities for building, using, and maintaining databases.

D

DASD. Direct access storage device.

data mark. Synonymous with DATAIN/DATAOUT attribute characters in ISPF.

data structure. In SDF II, a structure that is part of a panel. For output, it describes how data is provided by

the application. For input, it describes how data is presented to the application.

DBCS. Double-byte character set.

DCF. Document composition facility.

DCSS. Discontiguous saved segment.

device field (DFLD). In IMS/MFS the smallest area in a device input or output format block whose content and structure are defined by the user.

device list. A list of compatible device types. It is defined by the system programmer.

device table. Synonym for device type table.

device type. In SDF II, the name of a device or of a device list.

device type editor. An editor used for creating and maintaining the device type table.

device type table. A table containing the names of all device types supported by SDF II together with the features available on the devices. It is maintained by the systems administrator.

DFLD. Device field.

dialog. (1) The interaction between a user and a computer. (2) In SDF II, one or more panels and associated logic that establish an interactive session between SDF II and a user. A dialog prompts the user to enter information appropriate to the function requested and displays the results.

discontiguous saved segment (DCSS). (In VM/SP discontiguous shared segment.) An area of virtual storage outside the address range of a virtual machine. It can contain read-only data or reentrant code. It connects discontiguous segments to a virtual machine's address space so programs can be fetched.

discontiguous segment. In VM/370, a 64KB segment of storage that was previously loaded and saved and assigned a unique name. The segment (or segments) can be shared among virtual machines if it contains reentrant code. Discontiguous segments used with CMS must be loaded into storage at locations above a user's CMS virtual machine, and are attached when needed and detached when no longer needed.

DLIB. Distribution library.

double-byte character set (DBCS). A set of characters in which each character is represented by 2 bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets. Because each character requires 2 bytes, the typing, display, and printing of DBCS characters requires hardware and programs that support DBCS.

DSECT. Dummy control section.

dynamic area. In SDF II, an area that is filled with text at run time by the application program.

E

EBCDIC. Extended binary-coded decimal interchange code. A coded character set consisting of 8-bit coded characters. (A)

emphasis class. In SDF II, a set of predefined attributes. Emphasis classes can be specified for fields, marks, and attribute descriptors.

EXEC. An executable procedure that contains CMS or TSO commands and execution control statements, such as branches.

extended architecture (XA). An extension to System/370 architecture that takes advantage of continuing high performance enhancements to computer system hardware.

extended attribute. Any one of the color, highlight, programmed symbol set, outlining, mixed, or validation attributes.

extended binary-coded decimal interchange code (EBCDIC). A coded character set of 256 8-bit characters.

extended external source format. In SDF II, an extension of CSP/AD's external source format representing certain properties of CICS/BMS and IMS/MFS. See external source format.

externally controlled library. In SDF II, a library that is controlled by a library system other than ISPF/PDF Software Configuration and Library Manager (SCLM). Communication with the library system is by means of user exit routines. See controlled library.

external source format. CSP/AD's external source format is a commonly used means of representing applications and panels in an AD/Cycle framework. The format consists of a readable syntax of mark-up tags and attributes.

F

field attribute. A defined characteristic of a field, such as protected or unprotected, alphanumeric or numeric, detectable or nondetectable, displayable or nondisplayable, or intensity. See presentation attribute and inherent attribute.

field format. A field property that determines the character set that can go into a given field.

FMID. Function modification identifier.

format. A format is part of a panel. It defines how data appears on a screen. For output, it defines how data is presented on a screen. For input, it defines how data is entered on a screen by a user. A format may consist of different definitions for different device types. These definitions are called format instances.

format element. A part of a format, such as a variable field, a constant field, a dynamic area, a graphic area, a repeat format, or an include panel.

format instance. A part of a format that defines the appearance of data for a particular device type.

format mode. One of the four modes in which SDF II can display the layout of a panel. In this mode, marks show the extent of fields and areas. Contrast with initial value mode, name mode, and sample value mode.

G

GDDM-IMD. Graphical Data Display Manager — Interactive Map Definition.

generation. In SDF II, a process by which objects are created for use in the target systems or for prototyping the application.

graphical data display manager (GDDM). A group of routines that allows pictures to be defined and displayed procedurally through function routines that correspond to graphic primitives.

graphic area. An area that is reserved for later insertion of graphics by the application program.

H

horizontal array. An array that is read from left to right and line by line. For example:

choice (1) choice (2)
choice (3) choice (4)

See array and vertical array.

import. In SDF II, a process by which objects are imported into SDF II from one of the supported target systems, from SDF/CICS, or from an external source format structure.

I

IMS/MFS. Information Management System/Message Format Service.

include panel. A panel that is included in another panel. Examples are headers and trailers.

Information Management System/Virtual Storage (IMS/VS). A database/data communication (DB/DC) system that can manage complex databases and networks. Synonymous with IMS.

inherent attribute. An attribute that can be defined for variable and constant field marks, and data marks. After the field is defined, inherent attributes cannot be changed.

initial mode. In SDF II, one of the four modes in which SDF II can display the layout of a panel. In this mode, the Format window shows each initial value in its variable field. Contrast with format mode, name mode, and sample mode.

initial value. A value the SDF II user assigns to a variable field. The application program displays this value at run time if no value has been provided by the application.

Interactive System Productivity Facility (ISPF). An IBM licensed program that serves as a full-screen editor and dialog manager. Used for writing application programs, it provides a means of generating standard screen panels and interactive dialogs between the application programmer and terminal user.

ISPF. Interactive System Productivity Facility.

ISPF/PDF. Interactive System Productivity Facility/Program Development Facility.

J

JCL. Job control language.

JES. Job Entry Subsystem.

MACLIB. A library that contains macros, copy files, or source program statements.

map specification library (MSL). A VSAM file in which user-defined, map-related objects and system objects are kept. It is used by SDF/CICS, CSP/AD, and GDDM-IMD.

mark. In SDF II, a character used to define a format element, such as a field or area, or to provide some editing function. Examples include area marks, character marks, separator marks, and spacer marks.

MID. Message input descriptor.

MOD. Message output descriptor.

MSL. Map specification library.

Multiple Virtual Storage (MVS). See MVS.

MVS. Multiple virtual storage. Implies MVS/370, the MVS/XA product, and the MVS/ESA product.

N

name mode. One of the four modes in which SDF II can display the layout of a panel. In this mode, the Format window shows the name of the variable field in the field. Contrast with format mode, initial mode, and sample mode.

national language translation table. A table containing all the national-language-dependent keywords.

NLS. National language support.

nonprogrammable terminal (NPT). In SAA Basic Common User Access architecture, a terminal attached to a host processor in which most of the user-interface functions are controlled by the host processor.

O

object. In SDF II a panel, panel group, partition set, AID table, or operator control table stored in an SDF II library.

Figure 102 shows the equivalents for these objects in the target systems.

Figure 102. SDF II objects and target system equivalents

SDF II Object	IMS/MFS	CICS/BMS [1]	GDDM-IMD	CSP/AD	ISPF
Panel	Format set	Map	Map	Map	Panel
Panel group		Map set	Map group	Map group	
Partition set	Partition definition block	Partition set			
AID table	PF key parameter of the DEV statement		AID table		
Control table	Operator control table				

[1] SDF/CICS uses the same terms as CICS/BMS

operational prototype. A simulation of an application program to test or review simple functions, such as simple data base access, scrolling, error reporting, and online help panels. For some application programs, an operational prototype may include some characteristics of a data base, including some program code or ISPF tables. The operational prototype is used to determine the needs of the user of the application program and to ensure that the application program meets those needs. See simulative prototype.

P

page. In SDF II, part of a format instance that corresponds to an IMS/MFS physical page.

panel. (1) The information that is displayed at any one time on the screen. (2) An SDF II object. It consists of formats, data structures, and various tables. Each panel has at least one format.

panel command. A command that affects a part of the panel, the whole panel, or the flow of SDF II. Panel commands are entered on the command line. They can be assigned to program function keys.

panel element. (1) An element of a panel as displayed in the “Define Panel Instances” dialog, which denotes one of the following:

- format,
- format instance,
- page, or
- data structure.

(2) A line in the “Specify Panel Elements” dialog. It is used by the panel construction utility to create one or more fields, panel text, or a repeat format on the panel to be constructed.

panel group. An object within &SDF. that contains a list of panel names and describes the properties of these panels.

panel group instance. A part of a panel group that describes the properties of the panel group for a particular device type.

partition. All or a portion of the screen. Data is presented within the partition through a viewport, which is defined when the partition is created.

partitioned data set (PDS). A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

partition set. An SDF II object that consists of a group of partitions designed to share the same screen.

partition set instance. A part of a partition set that describes the properties of the partition set for a particular device type.

PDS. Partitioned data set.

PL/I. A programming language that is designed for use in a wide range of commercial and scientific computer applications. (A)

presentation attribute. An attribute that defines how information is presented on the screen, such as highlighting and color. Contrast with application attribute.

Print Services Facility (PSF). The access method that supports the 3800 Printing Subsystem Models 3 and 8. PSF can interface either directly to a user's application program or indirectly through the Job Entry Subsystems (JES) of MVS.

prototype. See simulative prototype and operational prototype.

PSF. Print Services Facility.

PTF. Program temporary fix.

pull-down. In SAA Common User Access architecture, a list of choices associated with a choice on the action bar. A user selects a choice from the action bar and a pull-down menu appears.

pull-down choice. A textual item on a menu. A user selects a choice to work with an object in some way.

R

reference name. A 1- or 2-character name used by SDF II as a synonym for the name of a variable field.

repeat format. A rectangular part of the format that can be repeated down a panel. All instances of a repeat format must have the same variable fields at the same relative horizontal positions as in the source format.

Report Program Generator (RPG). A programming language specifically designed for writing application programs that meet common business data processing requirements.

Restructured Extended Executor (REXX). An interpretive language used to write command lists.

REXX. Restructured Extended Executor language.

RPG II. Report Program Generator II.

S

SAA. See Systems Application Architecture solution.

sample mode. One of the four modes in which SDF II can display the layout of a panel. In this mode, the Format window shows each sample value in its variable field. Contrast with format mode, initial mode, and name mode.

sample value. A value the SDF II user assigns to a variable field. SDF II displays this value when the panel is tested or during prototype simulation.

SCLM. Software Configuration and Library Manager.

scrollable area. The window in the main panel behind which a scrollable area format can be scrolled.

scrollable area format. A separate format used with a scrollable area.

SDF/CICS. Screen Definition Facility Customer Information Control System. An online application development tool used by application programmers to define or edit maps, map sets, and partitions for CICS/VS Basic Mapping Support.

SDF II. Screen Definition Facility II.

separator. In SDF II, a mark used to separate the length of a field, its name, and its mark.

SFS. Shared file system.

shift-in character (SI). A code extension character used to terminate a sequence that has been introduced by the shift-out character to make effective the graphic characters of the standard character set. (I) Contrast with shift-out character.

shift-out character (SO). In SDF II, a code extension character that substitutes, for the graphic characters of the standard character set, DBCS. Contrast with shift-in character.

SI. Shift-in character.

simulative prototype. A simulation of a series of panels used by an application program to test or review the primary flow of interactions between the application program and its users. The panels may display initial values and may accept data entered by a user. See operational prototype.

skeleton. An object used as a model when creating a new object.

skip after attribute. In SDF II, a presentation attribute that causes the cursor to skip to the next unprotected field when the field in which the cursor is located has been filled.

SMP/E. System Modification Program Extended.

SO. Shift-out character.

SSF. Software support facility.

spacer. In SDF II, a mark that positions information on lines during panel definition; it is typically used for centering.

specification object. Synonym for object.

System Modification Program Extended (SMP/E). An IBM licensed program used to install software and software changes on OS/VS1 and OS/VS2 systems. In addition to providing the services of SMP, SMP/E consolidates installation data, allows more flexibility in selecting changes to be installed, provides a dialog interface, and supports dynamic allocation of data sets.

Systems Application Architecture (SAA) solution. A set of IBM software interfaces, conventions, and protocols that provide a framework for designing and developing applications with cross-system consistency.

T

target system. A system under which the application using an SDF II panel can be run. For example, CICS/BMS or IMS/MFS.

TSO. Time Sharing Option.

U

user exit routine. A user-written routine that receives control at predefined user exit points. In SDF II, for example, it is a CLIST or an EXEC.

V

variable field. A field in which data may be changed by the application program or by the user. It has a character string, which can be empty, defined at run time as its contents. If no contents are provided at run time, the initial value, if defined at specification time, is taken as default instead. Contrast with constant field.

vertical array. An array that is read from top to bottom and column by column. For example:

choice (1) choice (3)
choice (2) choice (4)

See array and horizontal array.

VM. Virtual machine. Implies VM/SP, VM/HPO, VM/XA, or VM/ESA.

VS. Virtual storage.

VSE. Virtual storage extended.

VTAM. Virtual Telecommunications Access Method.

W

window. In SDF II, a rectangular part of the screen where scrollable data is displayed and can be manipulated.

SDF II publications

The SDF II Release 4 publications are:

SDF II Licensed Program Specifications, GH19-6115-4
Contains the product specifications and warranty information.
Audience: Data processing manager, system programmer.

Introducing SDF II Release 4 for MVS, GH19-8261-0
Summarizes the functions, uses, requirements, and advantages of SDF II.
Audience: Data processing manager, system programmer.

Designing Panels with SDF II, SH19-8212-0
Introduces SDF II to new users and explains how to define panels. It also explains advanced functions of SDF II to experienced users and can be used as a reference to the functions of SDF II.
Audience: System programmer, application programmer.

SDF II Administrator's Guide, SH19-8211-0
Describes how to customize SDF II on an MVS system. It also explains how to import objects into SDF II, how to set up libraries and work with them, how to run SDF II from batch, and how to identify and report faults in SDF II to IBM support personnel.
Audience: System programmer, application programmer.

SDF II Reference Summary, SX11-6088-3
Lists and explains SDF II line and panel commands. It also lists the main dialogs and functions of SDF II.
Audience: System programmer, application programmer, application user.

Related reading

This book refers to the following publications:

CSP/AD External Source Format Reference, SH20-6433

GDDM Installation and System Management , SC33-0152

CSP/AD External Source Format Reference, SH20-6433

GDDM Version 2 Diagnosis and Problem Determination Guide, SC33-0326

GDDM V2 Guide for Users, SC33-0327

GDDM V2.1 Application Programming Guide, SC33-0337

GDDM Interactive Map Definition, SC33-0338

ISPF ISPF/PDF V2 MVS Installation Customization, SC34-4019

ISPF V2 MVS Dialog Management Services, SC34-4021

ISPF V2 MVS Dialog Management Guide, SC34-4112

ISPF V2 MVS Dialog Management Services Examples, SC34-4113

ISPF ISPF/PDF V2 MVS Installation Customization, SC34-4117

ISPF V3 MVS dialog Management Guide, SC34-4213

ISPF V3 MVS Dialog Management Services Examples, SC34-4215

SMP/E Reference, SC28-1107

TSO/E V2 REXX Reference, SC28-1883

CICS/MVS V2.1.2 Application Programmer's Reference, SC33-0512

CICS/ESA V3.3 Application Programming Reference, SC33-0676

CICS/OS/VS Version 1 Release 7 Application Programmer's Reference, GX33-6047

CICS/VS V1 Application Programmers Reference Macro Level, SC33-0079

CICS/OS/VS V1 Application Programmer's Reference (Command Level), SC33-0241

ISPF Examples V4.1, SC34-4451

ISPF Software Configuration and Library Manager (SCLM) Developer's Guide, SC34-4469

ISPF Software Configuration and Library Manager (SCLM) Project Manager's Guide, SC34-4470

ISPF Software Configuration and Library Manager (SCLM) Reference V4, SC34-4471

ISPF User'S Guide V4.1, SC34-4484

ISPF Services Guide V4.1, SC34-4485

ISPF Dialog Developer's Guide and Reference V4.1, SC34-4486

VisualGen: Planning V1.1, SH23-6553

VisualGen User's Guide and Reference V1.1, SH23-6559

Installation Guide for VisualGen Developer and Application Generators, SH23-6567

CICS OS/2 V1.20 System and Application Guide, SC33-0616

IMS/VS V2 Message Format Service User's Guide, SC26-4181

Index

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Y

Communicating Your Comments to IBM

Screen Definition Facility II
Administrator's Guide
Release 4

Publication No. SH19-8211-00

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing a readers' comment form (RCF) from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:

  (+43 1) 21145 - 4490

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies.

Readers' Comments — We'd Like to Hear from You

Screen Definition Facility II
Administrator's Guide
Release 4

Publication No. SH19-8211-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	□	□	□	□	□

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	□	□	□	□	□
Complete	□	□	□	□	□
Easy to find	□	□	□	□	□
Easy to understand	□	□	□	□	□
Well organized	□	□	□	□	□
Applicable to your tasks	□	□	□	□	□

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? □ Yes □ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SH19-8211-00

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Vienna Software Development Laboratory
SDF Development
Department 00/174
Lassallestrasse 1
A-1020 Vienna
Austria

Fold and Tape **Please do not staple** Fold and Tape

SH19-8211-00

Cut or Fold
Along Line

Program Number: 5665-366

Printed in Denmark by IBM Danmark A/S

SH19-8211-00