IMS Application Development Facility II

SH20-6593-02

Installation Guide

Version 2 Release 2

IMS Application Development Facility II

SH20-6593-02

Installation Guide

Version 2 Release 2

Note!

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

Third Edition (July 1997)

This edition applies to Version 2 Release 2 of IMS/ESA Application Development Facility II, 5665-348, and to any subsequent releases until otherwise indicated in new editions or technical newsletters. 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 below.

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

IBM Corporation, Department ICE
P.O. Box 49023
San Jose, CA, 95161-9023
United States of America

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

Contents

Figures

Notices

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

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

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling (1) the exchange of information between independently created programs and other programs (including this one) and (2) the mutual use of the information that has been exchanged, should contact:

IBM Corporation
555 Bailey Avenue, W92/H3
P.O. Box 49023
San Jose, CA 95161-9023

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

Programming Interface Information

This publication is intended to help customers do customization of IMS/ESA Application Development Facility II (IMSADF II). This publication documents information that is Diagnosis, Modification or Tuning Information provided by IMSADF II.

Attention: Do not use this Diagnosis, Modification or Tuning Information as a programming interface.

Trademarks

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

AT	OS/390
DB2	RACF
DATABASE 2	System/370
IBM	System/390
IMS/ESA	VTAM
MVS	

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Preface

This book is a guide to the processes needed to complete the installation and customization of IMS/ESA Application Development Facility II (IMSADF II) Version 2 Release 2.

The initial installation of IMSADF II requires that you use the *Program Directory for IMS Application Development Facility II Version 2*, which is part of the IBM Software Distribution package sent to IMSADF II customers.

This book assumes that any subsystems under which you want IMSADF II to execute (IMS, ISPF, or DB2) are already installed.

Organization

This book contains the following chapters:

Chapter 1, "Introduction," introduces IMSADF II post-installation and customization steps.

Chapter 2, "Preparing the IMSADF II Environment," describes the tasks you must do before customizing IMSADF II.

Chapter 3, "Customization Macros (DEFADF and ADFMFS)," describes the keywords for these two customization macros.

Chapter 4, "Building the Customization Jobs After Initial Installation," illustrates use of the ISPF Installation Dialog panels to complete installation.

Chapter 5, "Running the Customized Installation Jobs," lists step-by-step procedures for running the jobs you have customized through the dialogs. This completes installation. This chapter also describes each installation job.

Chapter 6, "Customizing Existing IMSADF II Systems," describes how to change IMSADF II options after IMSADF II is installed.

Chapter 7, "Performance Considerations," discusses tuning techniques to achieve maximum performance for IMSADF II.

Chapter 8, "Multiple IMSADF II Systems," discusses why you might have another IMSADF II system in your environment and the SMP decisions involved.

Chapter 9, "National Language Support," describes how to install screens and messages (including those defined by you) in languages other than English.

Chapter 10, "Post-SMP Maintenance," outlines steps you must take when applying maintenance to your IMSADF II system.

Chapter 11, "Sample Problem," provides a sample procedure you can use to verify IMSADF II's online operation.

Chapter 12, "Installation Macro Messages," lists the messages you may receive from the installation macro generation process.

Appendix A, "Sample IMSGEN Control Statements," shows the IMSADF II requirements for your IMS TM system.

Appendix B, "Sample Problem Examples," contains sample program code, including the SPA layout.

Appendix C, "DL/I Space Allocation Calculations," shows how to calculate DASD space requirements.

Appendix D, "SPA Size Calculations," shows how to calculate the amount of working storage required for various IMSADF II transactions.

Appendix E, "DB2 Space Allocation Calculations," shows how to calculate space requirements for dynamic rules databases in the DB2 environment.

Appendix F, "Source Members Requiring Special Handling," lists the members of IMSADF.ADFSLIB and IMSADF.ADFMAC requiring customization.

Appendix G, "MFS Naming Conventions for Supplied Screens," shows MFS naming conventions for the screens that are supplied with IMSADF II.

Appendix H, "Currency Requirements for Skipped Releases," explains what to do if you skipped a release of IMSADF II and are installing Version 2 Release 2.

Corequisite Publications

- *Program Directory for IMS Application Development Facility II Version 2* contains specific installation specifications and instructions.
- *SMP/E Reference*, SC28-1107, contains information necessary to use SMP/E.

Related IMSADF II Publications

- *IMS Application Development Facility II Version 2 Release 2 General Information*, GH20-6591.
- *IMS Application Development Facility II Version 2 Release 2 User Reference*, SH20-6592.
- *IMS Application Development Facility II Version 2 Release 2 Application Development Reference*, SH20-6594.
- *IMS Application Development Facility II Version 2 Release 2 Application Development Guide*, SH20-6595.
- *IMS Application Development Facility II Version 2 Release 2 Rules Documentation User's Guide*, SH20-6596.
- *IMS Application Development Facility II Version 2 Release 2 Data Dictionary Extension User's Guide*, SH20-6597.
- *IMS Application Development Facility II Version 2 Release 2 Master Index*, SH20-6599.
- *IMS Application Development Facility II Version 2 Release 2 Introduction to Using the Interactive ADF*, SH20-6601.

- *IMS Application Development Facility II Version 2 Release 2 Interactive ADF Administration Guide*, SH20-6602.
- *IMS Application Development Facility II Version 2 Release 2 DATABASE 2 Application Specification Guide*, SH20-6603.
- *IMS Application Development Facility II Version 2 Release 2 Diagnosis Guide*, LY20-6401.

Summary of Changes

SH20-6593-02 – July 1997

Following is a summary of the technical changes to IMS/ESA Application Development Facility II for Version 2 Release 2.

- SMP installation information was moved to the program directory.
- CICS information was removed.
- Numerous editorial changes were made.

Chapter 1. Introduction

This book explains the customization processes for IMSADF II. The book takes you step-by-step through the following tasks:

Planning and Preparing for IMSADF II: The first two chapters of this book contain planning and preparatory information. You must prepare your environment to execute IMSADF II transactions. Chapter 2, “Preparing the IMSADF II Environment” describes the environment in terms of storage and data set requirements, preparation of the dynamic rules databases, and subsystem preparation.

When you install IMSADF II, you are expected to select IMSADF II processing options to tailor your environment. Chapter 3, “Customization Macros (DEFADF and ADFMFS),” describes the options from which you may choose.

Submitting Jobs: The step-by-step job submission process required to initially install IMSADF II is covered in the Program Directory supplied with the product tape.

Chapter 4, “Building the Customization Jobs After Initial Installation” shows you how to use the ISPF Installation Dialogs to complete installation of your IMSADF II system.

Chapter 5, “Running the Customized Installation Jobs” gives you the step-by-step procedures required to install IMSADF II using the ISPF Installation Dialogs. It also describes the installation jobs you can do.

Changing Options: You may want to change your original options after IMSADF II is installed. Chapter 6, “Customizing Existing IMSADF II Systems” gives directions for specific customization options using the ISPF Installation Dialogs.

Achieving Optimum Performance: Chapter 8, “Performance Considerations” describes techniques for getting the best performance from your IMSADF II system.

Installing Multiple Systems: Chapter 7, “Multiple IMSADF II Systems” discusses the reasons for installing multiple IMSADF II systems. This chapter also discusses whether you should share SMP data sets among multiple systems.

Supporting Other Languages: National Language Support (NLS) is included in IMSADF II for non-English-speaking countries. You can see IMSADF II screens and messages in your own, as well as other, languages. Chapter 9, “National Language Support” describes the level of support offered and tells you how to install languages.

Maintaining IMSADF II: Chapter 10, “Post-SMP Maintenance” describes the special considerations involved in maintaining IMSADF II.

Testing IMSADF II: When IMSADF II is completely installed based on your needs, test the IMSADF II system using the application described in Chapter 11, “Sample Problem.” Testing helps you verify that you have correctly completed IMSADF II installation.

Diagnosing IMSADF II: Chapter 12, “Installation Macro Messages” lists the messages you can expect to receive from the installation process.

Note: If you want to use the Interactive Application Development Facility (IADF), start with the *IMS Application Development Facility II Version 2 Release 2 Interactive ADF Administration Guide* after you complete IMSADF II product installation using this book.

IMS Environment

IMSADF II executes as an IMS application program and must be installed in an IMS DB/TM environment. The IMSADF II system includes:

- IMS control blocks (DBDs, PSBs, ACBs, and FORMATs)
- IMSADF II system libraries
- IMSADF II dynamic rules databases

After the installation procedure is completed, IMSADF II can execute in supported IMS environments as:

- A message processing program (MPP) application
- A batch message processing (BMP) application
- A batch application, using the batch driver

The sample online application supplied with IMSADF II uses the IMS-supplied database DI21PART, which must be available in your IMS online environment.

Unique ADFID Libraries

In addition to the IMSADF II installed libraries, the ADFID libraries described below are needed to execute and maintain IMSADF II transactions.

ADFID libraries were assigned a high-level qualifier of NEWADF by the installation process (see “DEFADF Macro (Defining Basic IMSADF II Characteristics)” on page 15).

IMSADF.JCLLIB	Contains the jobs and procedures for installing, maintaining, and using IMSADF II.
IMSADF.RULES.SOURCE	Contains source statements that are input to an IMSADF II utility, the Rule Generator, from which IMSADF II applications are created. Also contains data for the sample problem.
IMSADF.DBDSRC	Contains source for DBDs supplied by IMSADF II.
IMSADF.PSBSRC	Contains source for PSBs supplied for IMSADF II and sample transactions.
IMSADF.SCREENS	Contains installation screen source statements, previously created by the Rule Generator. These statements can be MFS source. This library is input to the MFS utility.
IMSADF.RULLIB	Contains load modules called *rules*, previously created by the Rule Generator, which are used by execution time IMSADF II modules as logic

for your transactions. This library must be included in the STEPLIB concatenation for your IMS message region.

This library also contains intermediate load modules for user audit exits, DL/I exits, and special processing routines.

IMSADF.COMPLIB Contains composite rules load modules, output of the Rule Generator, which are used by IMSADF II modules as logic for your transactions. If you elect not to use composite rules load modules, you can delete this library. If you want to use this library, include its name in your IMS message region under the COMPMDD DD statement.

Other Libraries

Certain DL/I utilities and screen generations are executed during installation, and their output is directed to the following IMS libraries:

IMSVS.PSBLIB
IMSVS.DBDLIB
IMSVS.ACBLIB
IMSVS.FORMAT

IMSADF II Dynamic Rules Databases

This section describes the DL/I and DB2 support for the dynamic rules databases.

DL/I Support for IMSADF II Dynamic Rules Databases

The following IMSADF II dynamic rules databases are created and loaded during installation:

MFDPAR01 ????AUDT Audit database

MFDPSP01 ????SIGN Sign-On Profile database

MFDPMS01 ????MSGS Message database

MFC1WORK ????WORK Work database (IMS environment only)

The Work database is used if you elect not to use the IMS scratch pad area (SPA) for your IMSADF II conversational transactions. See “IMS SPA and Work Database Considerations” on page 13 for additional information.

Note: The question marks are replaced by the ADFID.

IMSADF II dynamic rules databases are accessed using HDAM and OSAM.

DB2 Support for IMSADF II Dynamic Rules Databases

When you choose DB2 support, the following IMSADF II tablespaces are created and loaded during installation:

TSAUDIT Audit database

TSSIGN Sign-On Profile database

TSMSG Message database

TSSPA Work database (IMS environment only)

The Work database table is used in an IMS environment for your IMSADF II conversational transactions. See “IMS SPA and Work Database Considerations” on page 13.

Tablespace names cannot be modified.

Chapter 2. Preparing the IMSADF II Environment

This chapter describes what you must do before customizing IMSADF II. Some requirements relate to IMS, while others relate to the database access method you choose for the dynamic rules databases (DB2DB parameter). “DEFADF Macro (Defining Basic IMSADF II Characteristics)” on page 15 explains the options. Some of the requirements apply regardless of the choices you make.

This chapter describes:

- Additional storage requirements for IMS
- Space requirements if you modify the installation procedure so that it doesn't use existing subsystem libraries
- Dynamic rules databases, whether DL/I or DB2
- Additional preparations for IMS administrators
- Modifying skeletal JCL
- SPA options

IMS Storage Requirements

Additional storage requirements depend on your selection of values for the following installation options (described in “DEFADF Macro (Defining Basic IMSADF II Characteristics)” on page 15). All areas except RULESVS are in storage acquired using a GETMAIN. RULESVS is in virtual storage.

WRKAREA Add the value of WRKAREA

RULESBL Multiply the supplied RULESBL value by 66 (the number of bytes in a BLDL entry)

COMPMBL Multiply the supplied COMPMBL value by 66 (the number of bytes in a BLDL entry)

RULESVS Add the value of RULESVS (in K bytes) to the storage requirements.

Space Requirements for Subsystem Libraries

IMSADF II uses the following existing subsystem libraries for control blocks and screen generation:

IMSVS.DBDLIB
IMSVS.PSBLIB
IMSVS.ACBLIB
IMSVS.FORMAT (for screens)

Note: The following allocations assume an average block size of 6144.

Figure 1. Additional Space Required for IMSADF II Components

DATA SET NAME	BLOCKS REQUIRED	DIRECTORY BLOCKS
IMSVS.PSBLIB	6	2
IMSVS.DBDLIB	2	1
IMSVS.ACBLIB	14	1
IMSVS.FORMAT	40	12

IMSADF II Dynamic Rules Databases Considerations (IMS and DB2)

This section describes database space allocations and considerations. The following IMSADF II dynamic rules databases (supplied with the product) can be either DL/I databases or DB2 databases, based on installation options:

- Audit database
- Message database
- Sign-on Profile database

Considerations specific to DL/I and DB2 are described in the following subsections.

Figure 2 shows the initial space allocation for the dynamic rules databases by type of database. Requirements are shown in blocks for DL/I or kilobytes (multiples of 1024) for DB2. PRImary and SECondary allocations are listed.

Space is allocated to the IMSADF II dynamic rules databases based on distributed system requirements, regardless of whether the databases are DL/I or DB2.

Figure 2. IMSADF II Database Space Allocations

DATA SET NAME	BLOCK SIZE	PRI	SEC	DL/I or DB2
IMSADF.MFDPAR01.DB	4096	180	60	DL/I
IMSADF.MFDPSPO1.DB	4096	60	60	DL/I
IMSADF.MFDPMSO1.DB	4096	180	60	DL/I
TSAUDIT	4096	780	260	DB2
TSMSG	4096	780	260	DB2
TSSIGN	4096	260	260	DB2
TSSPA	4096	1600	300	DB2

Chapter 4 of *IMS Application Development Facility II Version 2 Release 2 Application Development Reference* describes the format and structure of DL/I and DB2 databases in detail.

The direct-access storage requirement for IMSADF II databases depends on the number of segments or rows needed in each physical database to store IMSADF II definitions. Since you will be adding data to the dynamic rules databases as you develop applications, you should review your installation's needs and adjust space requirements accordingly. Appendix C, “DL/I Space Allocation Calculations” and Appendix E, “DB2 Space Allocation Calculations” can help you make those calculations.

The High Level Audit Language (HLAL) compiler can generate a static audit load module directly from source, as opposed to creating low level audit rules that are subsequently added to the Audit database. If your installation uses static audit load module generation, which gives better transaction performance, you reduce or eliminate additional space requirements on the Audit database outside of the distributed allocation.

DL/I Considerations

The IMSADF II dynamic rules databases are set up to use the HDAM access method with OSAM data sets. Review the parameters supplied for DBD generation and the disk storage space allocated for the dynamic rules databases to make sure they are consistent with the number of database records your installation anticipates for its databases.

The supplied DBD parameters and associated space allocations are sufficient for a new IMSADF II installation. Make initial and periodic reviews of these parameters to ensure that the specified values are adequate for current application requirements. Figure 3 shows the DBD parameter specifications that are allowed.

Figure 3. DBD Parameters

Database	Parameter Specifications Allowed
Audit	1,082 database records with an average database record size of 515 bytes. This is accommodated by using 8 root anchor points (RAPs) per physical block and 171 4K-physical blocks on three 3390 cylinders of primary allocation.
Message	1,503 database records with an average database record size of 360 bytes. This is accommodated by using 11 root anchor points per physical block and 167 4K physical blocks on three 3390 cylinders of primary allocation.
Sign-On Profile	136 database records with an average database record size of 1,356 bytes. This is accommodated by using 4 root anchor points per physical block and 57 4K-physical blocks on one 3390 cylinder of primary allocation.

Appendix C, “DL/I Space Allocation Calculations” can help you determine the adequacy of the space and DBD parameters for the dynamic rules databases at your installation.

The output from the calculations in Appendix C can be used to modify the DBDs in the IMSADF.DBDSRC library before running job ADFDBD of the installation procedure and/or the database space allocations in job ADFAALOC of the installation procedure.

If you use the Work database (????WORK) instead of the IMS SPA for the IMSADF II work area, select an allocation that fits your environment. Appendix D, “SPA Size Calculations” helps you make the necessary calculations. See “IMS SPA and Work Database Considerations” on page 13.

DB2 Considerations

The high level node for the SYSADF tables must be specified differently for each ADFID you have installed. DB2 requires that no two tables or indices have the same name.

Installation member ADFTBLD lets DB2 define the database and tablespaces. Appendix E, “DB2 Space Allocation Calculations” lists the tables in the logical equivalent of the dynamic rules databases:

Audit	3 tables
Message	7 tables
Sign-on Profile	4 tables

All of the related tables are contained in the same tablespace. Tablespace names are not customized based on options on the ISPF Installation Dialogs panels. The table names have the prefix SYSADF, which you do specify.

Master Table Handler rules are supplied to access dynamic rules databases for transaction processing. These rules are customized during installation based on the SYSADF value.

You may prefer to do allocation using IDCAMS (see the DB2 library for instructions). If you want, you can also create separate tablespaces for each table; however, separate tablespaces are usually not necessary.

The reasons for allocating a table to its own tablespace relate to application concurrency and table size. Concurrency is not really an issue for the dynamic rules databases. The tables cannot be modified by user applications, whose access is at the read-only level. Dynamic rules databases are updated only by the IMSADF II batch driver, which can be scheduled at an off-peak time by operations. The only installations for whom concurrency might be a concern are those who use IMSADF II project and/or user message sending functions extensively.

A DB2 **table** is considered small if it occupies fewer than 20 pages. Each IMSADF II **tablespace** contains twenty or fewer pages at installation time. As far as expansion goes:

- The S4 table in the Sign-on Profile contains project message rows. If your installation does not use this function, there is little need to modify the initial allocation or split this tablespace. If you want to do project message sending, this is the only row that will be updated by a user application.
- Each table in the Audit database could be separated if you don't choose to use the static audit load module option of the HLAL.
- With the Message database:
  - User data is most frequently added to the M1 and M2 tables as application messages are developed.
  - The M7 table is the only one updated by user applications; updating occurs if user message sending is initiated. If you do not use this function, there is no need to increase the M7 allocation or separate it for concurrency purposes.
  - The M4 and M6 tables usually have few, if any, rows.
  - The likely candidates for separation for volume purposes, not concurrency, are M5 and M3, which could be expanded greatly if you choose to generate

a lot of screen or error message help, respectively. M5 has a large row size.

See the DB2 library for more information.

IMS Preparation

The IMS DATABASE, APPLCTN, TRANSACT, TERMINAL, SPAREA, and MSGQUEUE macros must be included in an IMS system definition before you can use IMSADF II online.

When the ISPF Installation Dialogs customization is completed, the DATABASE, APPLCTN, and TRANSACT names reflect your unique IMSADF II qualifier. Then you can perform the IMS stage I system definition. See the IMS library for a detailed discussion of the Stage I macros and their options. A sample of these macros is in Appendix A, “Sample IMSGEN Control Statements.”

Note: Other TRANSACT parameters (for example, SCHDTYP, PRTY, and PROCLIM) should be considered at this time for IMSADF II system transactions, as for any other IMS application.

Storage and Pools

The IMS online IMSADF II is a large user of PSB pool resources if you use the DL/I Work database instead of a large IMS SPA or DB2 table. If you use the SPA, make sure your conversational work area pool (CWAP) is large enough. Some IMS buffer pools must be made larger than the normal default values. See Chapter 8, “Performance Considerations” on page 88 for buffer pool performance suggestions. See the IMS library for information about IMS buffer pool requirements and calculations.

TERMINAL Macro

The IBM 3270 Information Display System family is supported for online use of IMSADF II. This type of display station is supported through IMS interfaces.

Certain features and options pertaining to IMSADF II should be mentioned. A 3279 color terminal should have a feature code (`FEAT=`) denoting the use of extended color support if there are other terminals of the same type (for example, 3270-Axx) for which you do **not** want color support. The designated code is used by the IMSADF II Rule Generator to produce MFS source statements. The installation-selected code must be identified to IMSADF II through the DEVCHRS parameter of the DEFADF macro.

An example illustrates the situation. The following naming conventions for DIF/DOF members of the FORMAT library are described in the IMS library:

- First character--device type code (DEVNAME for IMSADF II)
- Second character--device characteristics (DEVCHRS for IMSADF II)
- Third character--upper or lower representation of first character of FMT label (depending on DOF or DIF)
- Remaining characters--rest of FMT label

DEVNAME and DEVCHRS make the difference in DIF/DOF sets.

Assume the following sets of terminals:

```
DEVNAME=(A2,A2,A2)
DEVTYPE=(6,2,6)
DEVCHRS=(17,0,0)
```

The Rule Generator creates **three** sets of DIF/DOF pairs, two containing extended attributes (DEVTYPE=6) and one without them. **Two** DIF/DOF pairs are created by the MFS utility, as illustrated below:

1. A2 + 17 + NNNNNN (common name)
2. A2 + 0 + NNNNNN (common name)

To insure that the MFS utility output name is unique, code a FEAT=Fn (n can be from 1 to 10) for the third device for which you want a DIF/DOF set created, and use the same value in the DEVCHRS for that terminal, as is now shown correctly below:

```
DEVNAME=(A2,A2,A2)
DEVTYPE=(6,2,6)
DEVCHRS=(17,0,n)
```

The PAGDEL option should be specified for terminals to be used by IMSADF II in an interactive or conversational manner; using PAGDEL ensures correct handling of MFS paging requests.

IMSADF II lets you specify sets of terminal characteristics for online use that are saved permanently for you. Chapter 3, “Customization Macros (DEFADF and ADFMFS)” on page 15 tells you how to do this. Once IMSADF II is installed, “Changing Terminal Types in IMSADF II” on page 76 tells you how to modify IMSADF II to change terminal types or characteristics.

SPAREA Macro

The SPAREA macro defines the number and maximum size of the scratch pad area (SPA) for all IMS application programs, including IMSADF II programs. The size you specify with SPAREA must be at least equal to the size specified in the TRANSACT macro.

You must specify whether you want to use the IMS SPA or the Work database to hold needed data between conversational iterations. If you choose the Work database, you must still specify a SPA for all IMSADF II transactions; the SPA size would be 28 bytes and would usually be CORE.

For example, if you code the TRANSACT macro as:

```
TRANSACT CODE=????T01,SPA=(28,CORE,FIXED),MODE=SNGL
```

You would code SPAREA as:

```
SPAREA   CORE=(n,yyyy)
```

where:

`n` is the number of SPA areas in virtual storage

`yyyy` is the size of the largest SPA

With any other conversational transactions in the system, yyyy would probably be greater than the 28 bytes required for IMSADF II.

The SPA sizes for IMSADF II can be calculated using Appendix D, "SPA Size Calculations." Most supplied IMSADF II transactions need a SPA of 6000 bytes if the Work database is not used.

Therefore, if you code the TRANSACT macro as:

```
TRANSACT CODE=????T01,SPA=(6000,DASD,FIXED),MODE=SNGL
```

You would code SPAREA as:

```
SPAREA   DASD=(n,6000)
```

where:

n is the number of SPA areas held on DASD storage

6000 is the size of the largest SPA

MSGQUEUE Macro

The MSGQUEUE macro tells IMS the maximum IMSADF II logical record length. An example follows:

```
MSGQUEUE DSETS=(3390),RECLNG=(672,1728)
```

In this example, 1728 is required to handle the largest IMSADF II message. The minimum logical record length (the default value) is 672.

Modifying Skeletal JCL

Skeletons are provided in IMSADF.ADFSLIB for use in IADF and in installation. Some skeletons are distributed with hard-coded DL/I parameter lists. Review the skeletons to make sure these parameter lists are appropriate for your installation. Some modifications may be desirable depending on which release of IMS you are using.

These skeletons are listed below. They can be modified in your own library. Do this with great care and only if necessary, as these skeletons are used in IADF.

ADFDATAS Does a database load for IMSADF II databases (job ADFDATA).

ADFWORKI Scratches, allocates, and initializes the IMSADF II Work database in a job of the same name.

PROCALS Executes the High Level Audit Compiler. Dialog customization results in a procedure that is copied to a procedure library in job COPYPROC.

PROCBS Executes the IMSADF II batch driver to update IMSADF II databases. Dialog customization results in a procedure that is copied to a procedure library in job COPYPROC.

PROCRDCS Executes the Rules Document facility under the IMSADF II batch driver. Dialog customization results in a procedure that is copied to a procedure library in job COPYPROC.

IMS SPA and Work Database Considerations

The conversational transaction drivers of IMSADF II can operate in one of two modes. One mode uses only an IMS SPA to retain all data across transaction scheduling and user interactions. The second mode uses a small SPA and a Work database for data retention. (The small SPA must be used if you select DB2 access for the Work database.) Either mode can be used within an IMSADF II system. A combination of the two modes is not allowed. The Work database format can be either DL/I or DB2.

DL/I Work Database Considerations

Consider the following if you choose to use the SPA-only mode (mode 1):

- SPAs reside in the IMS message queues when conversational transactions are enqueued.
- The SPA size for a standard processing transaction must equal the largest work area size required by that transaction. This can be extremely large, and the amount of virtual storage required in the message queue could become a problem for performance reasons.

Consider the following if you choose to use the combination SPA/Work database mode (mode 2):

- Only a small SPA is required.
- The database activity to the Work database can be isolated to a single database buffer pool, since OSAM is the access method.
- The PSBW (PSB work area pool) value for the IMS control region needs to be increased by the size of the maximum Work database record (WRKSIZE x WRKSEGS) for each region in which IMSADF II transactions are scheduled.
- Only one IMSADF II user can be operating from a terminal, since the LTERM name of the terminal is used as the key for the Work database.
- The DEFADF option WRKREDU reduces the number of records written to the IMS log; Work database reduction necessitates extra reloading of static rules. Since there are performance trade-offs, see Chapter 8, “Performance Considerations” on page 88 and “DEFADF Macro (Defining Basic IMSADF II Characteristics)” on page 15 for more information.
- Avoid using the /HOLD command since the Work database of the held conversation may be destroyed.

The `SPA=` parameter of the TRANSACT macro defines the size and type of SPA for the transaction. Since all IMSADF II conversational transactions begin execution with the same transaction, ????T01, `FIXED` must be specified on all IMSADF II conversational transactions if it is specified for ????T01.

Mode 2 operation uses a 28-byte SPA. The size required for mode 1 is variable. Appendix D, “SPA Size Calculations” shows how to calculate the SPA size for mode 1. The maximum size of the work area is constrained by IMS. The size of a real SPA is limited. The size of a database segment is also limited (see the `BYTES=` parameter on the SEGM statement). See the IMS manuals for more information on these topics.

The number of levels in the Work database is defined at installation time and can be a maximum of 15. The first 8 bytes of the root segment contain the key of the database record. The maximum work area, therefore, is:

```
( (number of levels) * (segment size) ) - 8
```

The other segments are non-keyed segments and are accessed with the `F` (first) command code. Database records are deleted at termination and inserted at sign-on. The database record can expand to the maximum 15 segments from the initially inserted root segment, or to the maximum specified during the IMSADF II installation process. The conversational drivers are set up to access and replace the minimum number of segments.

A member is supplied in IMSADF.JCLLIB, ADFWORKI to allocate and initialize the Work database. Allocation of four cylinders of 3390-1 space is enough to support about 50 concurrent IMSADF II transactions. The supplied DBD for the Work database calls for 50 root segments in the root addressable area.

DB2 Work Database Considerations

If you choose DB2 access for the IMSADF II dynamic rules databases, you must use a small SPA and DB2 table combination. The table is defined as LONG VARCHAR. Your space allocations are based on the size of WRKAREA and the number of conversations you want to be active at the same time. See Appendix D, “SPA Size Calculations” for help in determining the WRKAREA value. You will need two rows per LTERM if WRKAREA is over about 32,000. The Work database table is allocated in the 32K buffer pool.

As a terminal operator enters a transaction, a row is either created or updated using LTERM as the key. If a transaction is properly completed, the row is deleted. If the user enters `/EXIT` or otherwise abnormally terminates, the row remains until the next time the same LTERM originates a transaction (at that point the row is deleted). DB2 tables are effectively sequential randomizers, so your space allocation should reflect more than the number of conversations active at one time.

Ideally, you should allocate enough space so that each LTERM executing IMSADF II transactions can be active, since this table is relatively small.

Chapter 3. Customization Macros (DEFADF and ADFMFS)

This chapter describes the DEFADF and ADFMFS customization macros. You may want to familiarize yourself with these macros before proceding with customization to get a basic understanding of the values you enter in the ISPF Installation Dialogs.

These macros define IMSADF II:

- Basic characteristics (DEFADF)
- Screen characteristics (ADFMFS)

Both macros are supplied with default values. You can change the values, as appropriate for your environment, or keep the defaults. This chapter describes all of the keywords used in both macros.

DEFADF Macro (Defining Basic IMSADF II Characteristics)

The DEFADF macro keyword values are shown in Figure 4.

Keyword values described in this chapter are in the form required when the macro is invoked. The descriptions in this chapter are intended to supplement the help panels where a more complete explanation is necessary.

```
Label   DEFADF ADFID=MFC1,                DETERMINED AT INSTALLATION  X
               ASMBLR=F,                                              X
               COMMLEN,                                               X
               DATEFMT=U,                                             X
               DB2DB=N,                                               X
               DELIM=$$,                                              X
               ENVIR=IMS,                                             X
               KANAME=STD,                                            X
               MCTID,                    IGNORE EX-CICS               X
               NCPRINT=PRINTER,                                       X
               RGLIB=L,                                               X
               RACF=TERMINAL,                                         X
               SPATS=,                   IGNORE EX-CICS               X
               V50RES=YES,                                            X
               MFSTRLR=1,                                             X
               TRXTRLR=,                                              X
               WRKSIZE=6000,                                          X
               WRKSEGS=5,                                             X
               WRKAREA=30000,                                         X
               WRKREDU=N,                                             X
               DEVFEAT=,                                              X
               DEVNAME=(2),                                           X
               DEVTYPE=(2),                                           X
               DEVCHRS=(0),                                           X
               TRMDFLT=1,                                             X
               USRLANG=E,                                             X
               ALTLANG=,                                              X
               ADFNODE='IMSADF',         DETERMINED AT INSTALLATION  X
               NEWADF='IMSADF',          DETERMINED AT INSTALLATION  X
               OLDADF='IMSADF',                                       X
               IMSNODE='IMSVS',                                       X
               IMSTEST='IMSVS',                                       X
               SMPNODE='ADFSMPE',        DETERMINED AT INSTALLATION  X
               CICNODE=,                 IGNORE EX-CICS               X
               RULESVS=0,                                             X
               RULESDD=,                                              X
               RULESBL=0,                                             X
               XAMANAGE=N
               COMPMDD=,                                              X
               COMPMOD=NO,                                            X
               COMPMBL=0
        END
```

Figure 4. DEFADF Macro With Supplied Values

DEFADF keywords are defined in alphabetical order below.

Note: Several keywords are interdependent with other keywords (indicated by a "See also" reference at the end of a keyword description). When this is the case, see the descriptions of all interdependent keywords as well.

Label Can be specified, but is ignored.

ADFID A four-character identifier of the IMSADF II system being defined. Each IMSADF II system must have a unique ADFID. Do **not** use the ADFID as an application system ID (its use is reserved for IMSADF II system transactions).

If you install IMSADF II on IMS SMP data sets, do **not** use an ADFID that could cause duplicate LMOD, SRC, or MOD entries (for example, do not use DFSn as an ADFID).

Before choosing your ADFID or if you want to establish multiple execution time load modules using the same base modules, see Chapter 7, “Multiple IMSADF II Systems.”

Note: This value was determined when IMSADF II was installed.

ADFNODE The high-level qualifier for those IMSADF II data sets that can be shared by all IMSADF II systems. The value you specify must be less than or equal to 22 characters.

Note: This value was determined when IMSADF II was installed.

ALTLANG The alternate language(s) in which you want screens and messages to appear to the terminal end user. See Chapter 9, “National Language Support,” for a complete discussion of use of national languages and instructions for installing a national language. To be able to build screens and retrieve IMSADF II system messages in the languages you support, you must enter a one-position value representing the desired language and associate it with a SYSID (not ADFID); use the SYSID by which you want supplied messages to be retrieved.

An optional third value is a comma followed by the literal DBCS (Double Byte Character Set). DBCS notation is used by certain Far Eastern languages when the character set exceeds 256. DBCS is the literal you use to indicate that two bytes are required internally to represent one byte of data on the screen.

The supplied language values and their definitions follow:

E English (stored internally as blank)
F French
G German
K Korean (requires DBCS)
J Japanese (requires DBCS)
P Portuguese
S Spanish
W Swedish

Enter multiple pairs of operands, separated by commas, within the parentheses, as shown in the following example:

```
ALTLANG=(P,PORT,J,KOBE,DBCS)
```

Here `PORT` and `KOBE` are the SYSIDs for messages in Portuguese and Japanese, respectively. Append the comma and DBCS keyword, if it is required.

Other examples of this parameter follow:

```
ALTLANG=(P,PORT,F,FREN)

ALTLANG=(P,PORT,J,KOBE,DBCS,F,FREN)
```

Be sure you specify SYSIDs that are **not** used in standard IMSADF II applications, or you will not know which messages come from the user and which are system messages. In addition, make sure each SYSID you name is unique in at least the first two characters, as these characters are used in the creation of format names. See Appendix G, “MFS Naming Conventions for Supplied Screens” on

page 192, to see how IMSADF II screens are named for the IMS environment.

If a language you need is not one of those listed above, you must use the USRLANG program. See “Installing User Languages” on page 102 for information about USRLANG.

ASMBLR The default Assembler program to be used by the Rule Generator. One of three values can be specified:

- **ASMBLR=F** Indicates that the standard OS/VS Assembler is to be used.
- **ASMBLR=H** Indicates that the Assembler-H program product is to be used.
- **ASMBLR=A** Indicates that the High Level Assembler (ASMA90) is to be used.

CICNODE Ignored.

COMMLEN The number of bytes (in decimal) that you want IMSADF II to reserve at sign-on in the SPA for the user Communication Area (SPACOMLN). Possible values are between 0 and the maximum size of the SPA, minus 800 bytes. The format is as follows:

COMMLEN=nnnn `nnnn` is 0 - (WRKAREA minus 800 bytes).

If nothing is entered, `COMMLEN=0`.

You must code a value only if you have a Lockword exit and want to place information from your exit in the Communication Area of the SPA.

Transaction processing may increase the value of SPACOMLN. The total Communication Area is preserved if the user returns to any of the sign-on modules to view the Primary Option or Secondary Option Menus. When a project/group switch is made, however, SPACOMLN is reset to its value at sign-on.

Note: Since processing logically *terminates* in the project/group switch, it is not desirable to carry over residual transaction information. Carrying information over could also cause a transaction in the new project/group, which was already close to maximum WRKAREA size, to exceed the maximum.

By resetting SPACOMLN to the sign-on value, any residual information from the sign-on Lockword exit is carried forward.

There are application considerations that may apply if your installation has used transaction Communication Area processing in the past.

If you specify a COMMLEN, your application programmers may have to change their mapping of the Communication Area for existing transactions; this is because the area established at sign-on precedes that required at transaction driver time. In addition, the Communication Area length for the transaction may have to be changed to consider the sign-on COMMLEN. Any specification for COMMLEN readjusts the area to the MAXIMUM of the current and new values, not the sum of them. The following figure illustrates the layout:

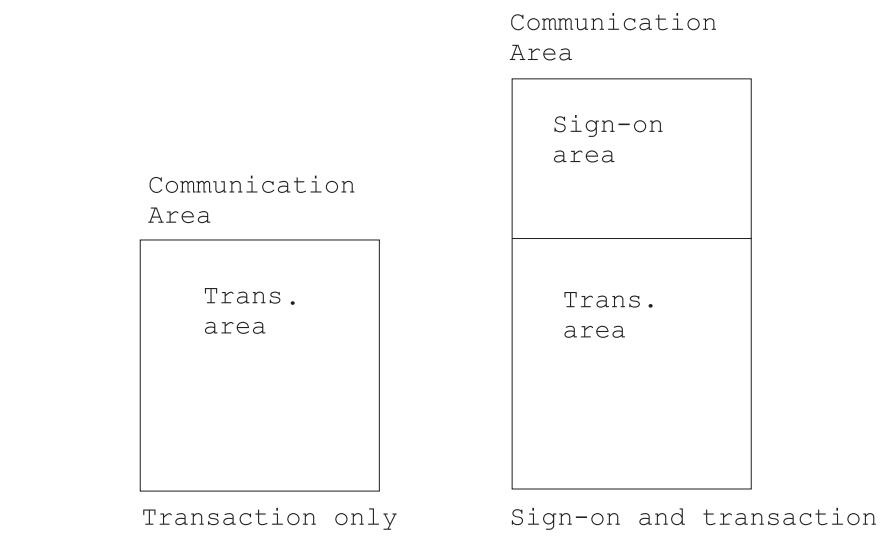

Figure 5. Communication Area User Mapping

See also the WRKAREA parameter.

COMPMBL Number of dynamic BLDL entries maintained for the composite rules load modules library (IMSADF.COMPLIB). The list is maintained in frequency-of-use sequence. Main storage requirements can be calculated by multiplying the number of entries to be maintained by 66, the length of a BLDL entry.

COMPMBL provides performance benefits for users of composite rules load modules. In a storage-constrained environment, specifying a large number for this parameter can cause excessive paging. At initialization, IMSADF II issues a GETMAIN for the amount of storage requested.

The best performance is realized when this parameter is used in conjunction with RULESVS.

If `COMPMOD=NO`, this parameter should not be specified (but defaults to zero).

See also the COMPMOD, RULESDD, RULESVS, and V50RES parameters.

COMPMDD Specifies the ddname for the IMS message region statement used to describe your composite rules load modules library (IMSADF.COMPLIB).

Use this option only if you choose `COMPMOD=YES` or `ALT`.

If you elect this option, IMSADF II goes **only** to this DD statement for all composite rules load module requests; this minimizes your library search time. If you do not specify a name, IMSADF II searches for composites through either the STEPLIB concatenation or RULESDD, depending on what option you select for RULESDD.

It is recommended that you specify this parameter, since IMSADF II checks for its presence at initialization time and goes through RULESDD/STEPLIB if the DD statement is missing; this degrades performance.

See also the COMPMOD and RULESDD parameters.

COMPMOD Indicates your use of composite rules load modules. See "Composite Rules Load Modules" on page 91 before selecting a value for this keyword.

Possible values are:

COMPMOD=NO (the default)
COMPMOD=YES
COMPMOD=ALT

If you choose COMPMOD=NO (the default), IMSADF II does not try to load a composite rules load module across your ADFID for nonconversational transactions (conversational transactions are selectable by TRXID). IMSADF II loads each rule required for an IMSADF II transaction as the need arises.

COMPMOD=YES indicates that a composite rules load module is to be used. It also tells IMSADF II that no further searching is to take place if the composite rules load module or the IMSADF II preload rule does not contain all the rules required for the transaction. If rules are missing, you receive a message and transaction processing is halted.

Specifying COMPMOD=YES lets you be certain that all composite rules load modules are complete and that maximum benefit is being gained from this performance option.

COMPMOD=ALT also indicates that a composite rules load module is to be used. This selection should be made if all rules required to execute a transaction are not present in the composite rules load module. If you choose this option, IMSADF II loads a composite rules load module. If individual rules are missing, they are retrieved as required. The RULESDD library, if requested and present, is searched. If RULESDD is not present in the message region JCL, STEPLIB is searched.

Notes:

- You must specify COMPMOD=ALT (if you want to use composites) if both Primary and Secondary Option Menu rules are not preloaded by IMSADF II for conversational processing.
- If you select ALT and are missing several rules, performance for the transaction may be similar to performance when NO is selected; this is because each missing rule is being loaded.
- The performance when ALT is specified is the same as that for YES **as long as** all rules are contained in the composite or are preloaded by IMSADF II.

It is recommended that you initially choose COMPMOD=ALT.

See also the COMPMDD, RULESDD, and COMPMBL parameters.

DATEFMT Specifies the display format of fields defined as TYPE=DATE.

Possible values are:

U mm/dd/yy (the default)
S yy-mm-dd
B yy mm dd
E dd.mm.yy
O dd/mm/yy
V mm/dd/yy or mm/dd/yyyy*
T yy-mm-dd or yyyy-mm-dd*
C yy mm dd or yyyy mm dd*
F dd.mm.yy or dd.mm.yyyy*
P dd/mm/yy or dd/mm/yyyy*

* = the 4-character year when the TYPE=DATE fields are specified with a length of 8.

DB2DB Specifies whether DL/I or DB2 is to be used for the supplied IMSADF II Audit, Message, and Sign-on Profile data.

Possible values are:

DB2DB=N Use DL/I databases (the default).
DB2DB=Y Use DB2 tables.

A value of N means that DL/I is to be used for the IMSADF II dynamic rules databases; Y means that DB2 tables are to be used for the supplied data.

This value **cannot** be changed after installation.

DELIM Two alphanumeric characters used as an end-of-text delimiter for IMSADF II batch driver input.

If nothing is specified, DELIM=$$.

The delimiter is used to terminate message and help input when it exceeds one line. Make sure that the values you select are not nor ever will be present in input to the batch driver. It is recommended that you take the default, unless you have a specific need to do otherwise.

DEVCHRS Enables the Rule Generator to build the correct FEAT= option on the MFS DEV statement for IMSADF II transactions on the Segment Display Screen. The following values represent sets of terminal characteristics that are named in the IMS TERMINAL macro FEAT= keyword:

DEVCHRS=(nn,nn,..)

The rest of this description applies only to the IMS environment.

The possible values for this keyword cover all currently valid parameters for FEAT. The values are listed in Figure 6 on page 22, along with the value that you code in DEVCHRS to represent them.

```
0    (IGNORE) (default)
1-10 (feature code F1 - F10 if used)

11   (NOCD,PFK,PEN)          12   (CARD,PFK,PEN)
13   (NOCD,NOPFK,PEN)        14   (CARD,NOPFK,PEN)
15   (NOCD,DEKYBD,PEN)       16   (CARD,DEKYBD,PEN)
17   (NOCD,PFK,NOPEN)        18   (CARD,PFK,NOPEN)
19   (NOCD,NOPFK,NOPEN)      20   (CARD,NOPFK,NOPEN)
21   (NOCD,DEKYBD,NOPEN)     22   (CARD,DEKYBD,NOPEN)
```

Figure 6. Numeric values for DEVCHRS (Representing IMS TERMINAL Macro Characteristics)

The values named first (up to the maximum specified in TRMDFLT) are carried as the default DEVCHRS; if you do not specify this value, the default is `DEVCHRS=0`. If you do **not** generate the same transaction id (TRXID) for both color and non-color transactions, code the value for `IGNORE(0)`.

Note: If you generate the same transaction for multiple terminals and DEVCHRS is not zero for **all** of them, then you must code the same DEVCHRS values for **each** terminal as was specified in your IMS stage I system definition.

IMSADF II normally generates a single MOD with the IGNORE operand on the SOR statement. If multiple terminals are specified, multiple DIF/DOF pairs are generated. If any features are specified in the DEVCHRS, MOD cannot be generated as IGNORE, and the DEV statements must specify the exact characteristics stated for the TERMINAL macro in an IMS system definition.

For more information, see the descriptions in the IMS library of the `FEAT=` parameter of the TERMINAL macro and the IGNORE operand.

This parameter replaces DEVFEAT, which is used in the Rule Generator GENERATE statements.

Specify each desired set of characteristics within parentheses, separated by commas. For each set you code, you must also code its corresponding DEVNAME/DEVTYPE value.

See also the DEVNAME, DEVTYPE, and TRMDFLT parameters.

DEVFEAT No longer used and is ignored if specified.

DEVNAME Specifies the IMS identification characters of 3270 terminal types that can be used with IMSADF II.

Possible values are:

DEVNAME=(2) (the default)
DEVNAME=(2 ,Axx“ ,...“)
DEVNAME=(Axx ,Axx“ ,...“)

The DEVNAME operand value is `2` (for the 3277 display device model 2) or a two- or three-character ID that starts with `A`. `DEVNAME=(2)` is the default.

Specify each desired terminal type within parentheses, separated by commas. For each terminal name you code, you must also code its corresponding DEVTYPE and DEVCHRS parameters.

This parameter is used in MFS source generation for supplied IMSADF II transactions (for example, the sample problem and database administration functions). The identification characters must match the IMS SYSGEN selection for the appropriate device types. The Rule Generator creates an MFS device statement for each operand value.

Screen formats are duplicated for each device (see DEVTYPE for screen size considerations).

The first named values, up to the maximum specified in TRMDFLT, are carried as the default DEVNAME for the Rule Generator and are used to create screens for an application if there are no overriding values. All named values are used to create screen formats for the supplied transactions.

See also the DEVCHRS, DEVTYPE, and TRMDFLT parameters.

DEVTYPE Specifies the screen size and characteristics that can be used with the transactions supplied with IMSADF II.

Possible values are:

```
DEVTYPE=(2) (the default)
DEVTYPE=(n,n,...)
```

where `n` can have the following values:

Value	Meaning
2	3270; screen size of 24 lines by 80 columns
3	3270; screen size of 36 lines by 80 columns
4	3270; screen size of 43 lines by 80 columns
5	3270; screen size of 27 lines by 132 columns
6	3279; screen size of 24 lines by 80 columns (color)
7	3279; screen size of 32 lines by 80 columns (color)
8	3290; screen size of 62 lines by 160 columns
9	5550; screen size of 24 lines by 80 columns (DBCS)
10	3270; screen size of 43 lines by 80 columns (color)

Notes:

1. The supplied transactions do not use extended color features; however, default colors appear when a DEVTYPE of 6, 7, or 10 is specified.
2. The IBM 5550 is a Japanese DBCS terminal.

Each terminal DEVTYPE operand must have corresponding DEVNAME and DEVCHRS operands.

The first named values, up to the maximum specified in TRMDFLT, are carried as the default DEVTYPE for the Rule Generator and are used to create screens for an application in the absence of any overriding values. All named values are used to create screens for the supplied transactions.

See also the DEVNAME, DEVCHRS, and TRMDFLT parameters.

ENVIR The subsystem environment for this ADFID.

The only value is:

ENVIR=I IMS

This value **cannot** be changed.

IMSNODE The high-level qualifier for IMS system data sets (for example, IMSVS.RESLIB) other than the DBD/PSB libraries. IMSNODE must be less than or equal to 22 characters.

See also the IMSTEST parameter.

IMSTEST The high-level qualifier for the IMS data sets used for DBDs and PSBs. The initial default is the same as the initial value for IMSNODE. If you have a different node name for DL/I control block libraries (a name other than RESLIB), this value provides additional flexibility. This value can be useful in testing. It must be less than or equal to 22 characters.

See also the IMSNODE parameter.

KANAME The default naming conventions to be used by the Rule Generator and the high-level audit language for key audits.

Possible values are:

KANAME=STD (the default)
KANAME=ALT

If `KANAME=STD`, the first eight characters of key audit keys are KEYAUDIT. If `KANAME=ALT`, the first eight characters are `ssssnnnn` where `ssss` is the application system ID and `nnnn` is the IMSADF II audit group code.

Note: This value is overridden if it is coded on Rule Generator input statements during development of transactions.

MCTID Ignored.

MFSTRLR A single, non-blank character used to create unique names for members in the IMSVS.FORMAT library (its value is **not** related to the last character of ADFID).

Suppose you have two IMSADF II systems named MFC1 and MFC2. If these two systems are essentially the same, both systems can use the same set of screens. Therefore, they can be given the same MFSTRLR value. However, if the systems are different (for example, MFC1 is a production system, MFC2 is a test system, and both MFC1 and MFC2 have different screens), their MFSTRLR values must be different.

For information on use of MFSTRLR in an environment with multiple IMSADF II systems, see Chapter 7, “Multiple IMSADF II Systems.”

NCPRINT Establishes a default printer LTERM name that is to be used by all nonconversational transactions running under this ADFID. This default name is used when the `LTERM=` parameter is not specified on the BUILDT08 statement in a nonconversational PSBGEN. Enter the LTERM name you select.

NEWADF The high-level qualifier for IMSADF II data sets that are unique to the ADFID being defined (for example, IMSADF.PSBSRC). The qualifier must be less than or equal to 22 characters.

Note: This qualifier was determined when IMSADF was installed.

OLDADF The high-level qualifier for any previously-existing IMSADF II system database. The qualifier must be less than or equal to 22 characters.

This name is used only when the format of an IMSADF II database has changed. Otherwise it is ignored.

RACF Determines the origin of the IMSADF II userid for the purposes of sign-on verification in online conversational transactions.

Possible values are:

TERMINAL Use the value entered on the standard sign-on screen as the userid. This is the default.

RACFUSER Use the first six characters of the RACF userid, which is located in the IOPCB as the IMSADF II userid.

RULEGEN Use the normal sign-on userid unless there is a special literal on the input format; in that case, use the RACF userid (first 6 characters). The MID literal can be specified at sign-on screen creation (Rule Generator operand `SIGNON=RACF`) and appears in place of the userid data. This option can be used during your conversion period to change from the TERMINAL form to the RACFUSER form.

Either of the last two options can eliminate the need for terminal end-users to enter the userid twice, at RACF sign-on and IMSADF II sign-on. In addition to this option, the sign-on screen for the SYSID must be generated with `SIGNON=RACF` (see *IMS Application Development Facility II Version 2 Release 2 Application Development Reference*).

If you select RACFUSER, the following sign-on profile considerations apply:

- The supplied input for the dynamic rules databases, IMSADF.JCLLIB(TRANSIN), contains sign-on profile data for userid 999999, which lets this user execute Database Administrator (DBA) transactions (which maintain the dynamic rules databases) and the sample application (SAMP) for installation verification.
- Add the correct userid(s) for the DBA transaction to the database **before** implementing RACFUSER; otherwise you cannot access the DBA transaction (unless 999999 is a valid RACF userid). Add your own userid to that of SAMP so you can verify the IMSADF II installation, as described in Chapter 11, “Sample Problem.”

The recommendation for full RACF implementation is:

1. **Initially** specify RULEGEN.
   The supplied sign-on screens do not specify the RACF literal, so the 999999 userid is valid.
2. Add the proper RACF userids to the Sign-On Profile database for the DBA transaction (and SAMP if you plan to use it).
3. Have developers generate RACF sign-on screens.
4. Change RULEGEN to RACFUSER after installation (see Chapter 6, “Customizing Existing IMSADF II Systems”).

Note: IADF users who test with BTS have a control statement added that sets the userid field in the IOPCB to 999999; this lets them execute supplied transactions.

RGLIB Denotes the form of the library where the Rule Generator can find user exits (that is, Audit exits, DL/I exits, and special processing routines).

This parameter must only be specified or defaulted as:

RGLIB=L

This value indicates to the Rule Generator where it can find exits when it is generating the transaction driver for an IMSADF II application. During this processing, the Rule Generator creates linkage editor INCLUDE statements for IMSADF II-supplied exits, and for exits that you specify in your Rule Generator input. The Rule Generator expects to find these exits as load modules in the SYSLIB concatenation.

Existing object text exits are not supported and must be converted to load module form after installation by link-editing them into RULLIB.

The installation process puts IMSADF II-supplied exits in IMSADF.ADFLOAD for you. Later, IMSADF II application developers must compile any exits that they write and link them to IMSADF.RULLIB. Both libraries are in the SYSLIB concatenation (from which the Rule Generator INCLUDEs them).

Note: Expect a return code of 4 from your link-edit, since all references are not resolved until you invoke the Rule Generator.

RULESBL Number of dynamic BLDL entries maintained by IMSADF II for the rules load modules library (IMSADF.RULLIB).

RULESBL=0 (is the default)

The list of BLDL entries is maintained in frequency-of-use sequence. Main storage requirements can be calculated by multiplying the number of entries to be maintained by 66, the length of a BLDL entry.

RULESBL provides performance benefits by eliminating BLDLS for frequently referenced composite rules. In a storage-constrained environment, specifying a large number for this parameter can cause excessive paging. At initialization, IMSADF II issues a GETMAIN for the amount of storage requested.

Best performance is realized when this parameter is used in conjunction with RULESVS.

See also the RULESDD, RULESVS, and V50RES parameters.

RULESDD Ddname for the IMS message region statement used to describe your rules load modules library (IMSADF.RULLIB). If you elect this option, IMSADF II accesses this DD statement in your message region for rules requests, to reduce your library search time, unless you specify COMPMDD. In that case, all requests for composite rules load modules are satisfied through the COMPMDD library **only**.

Note: If you use composite rules load modules (`COMPMOD=YES` or `COMPMOD=ALT`) and you do not specify COMPMDD, the RULESDD library is searched for the composites.

If you do not specify a name, IMSADF II searches for rules through the STEPLIB concatenation.

It is recommended that you specify this parameter; IMSADF II checks for its presence at initialization time and goes through STEPLIB if the DD statement is missing, degrading performance.

See also the COMPMDD parameter.

RULESVS Amount of virtual storage in `nK` bytes, up to a maximum of 9999K, which is to be available to dynamically manage rules loaded below the 16 megabyte line. Enter the numeric value of K storage. The default is 0.

IMSADF II manages RULESVS storage in frequency-of-use order to keep the most active rules (composite or individual) in storage at one time. The storage specified must be available in each IMS message region in which IMSADF II transactions are processed. Maximum performance gains are realized when you use this option in conjunction with RULESBL and COMPMBL to load rules. In a storage-constrained environment, specifying a large number for this parameter can cause excessive paging. See “Managing Rules Storage” on page 90 for more information.

If the default (0) is taken, a BLDL and LOAD are issued for each rule.

See also the COMPMBL, RULESBL, and V50RES parameters.

SMPNODE The high-level qualifier for SMP data sets. The initial default is the same as the initial value for ADFNODE. If you want IMSADF II to use the IMS SMP data sets, change SMPNODE to the IMSNODE value. SMPNODE must be less than or equal to 22 characters.

See also the IMSNODE parameter.

SPATS Ignore.

TRMDFLT Establishes the number of sets of DEVNAME/DEVTYPE/DEVCHRS that are maintained in ADFOPTNS as defaults. The maximum number of default terminal characteristics is five.

Specify a number from 1-5 (the default is 1).

Figure 7 shows an example.

These DEFADF operands:

```
TRMDFLT=2
DEVNAME=(A2,A4,A3,A3)
DEVTYPE=(2,4,7,3)
DEVCHRS=(0,17,3,11)
```

result in the following default terminal definitions:

1. A 3270-A2 with 24 lines and MFS `FEAT=IGNORE`
2. A 3270-A4 with 43 lines and `FEAT=(NOCD,PFK,NOPEN)`

The next two terminals have MFS created for supplied transactions, but are not defaults:

3. A 3270-A3 (3279-3B) with extended color support, (`FEAT=3`) and 32 lines
4. A 3279 Model 3B without color, but with `FEAT=(NOCD,PFK,PEN)`

Note: Unless overridden, one MID/MOD and two DIF/DOF pairs are created for each transaction.

Figure 7. Installation Specification Example and Results

Since the creation of screens for multiple terminal types results in a quickly expanding set of MFS DIF/DOF pairs, the IMSVS.FORMAT library can become very large. If you specify multiple terminal type defaults, the formats are created each time a transaction is generated without overrides. For this reason, it is **not** recommended that you specify all terminal types you think may ever be necessary, but only those that you use for **most** transactions.

You are better advised to select one or two sets of defaults, and override them specifically on the few occasions when you want to generate more. (The IADF user can set up a profile to handle this situation.)

See also the DEVNAME, DEVTYPE, and DEVCHRS parameters.

TRXTRLR A single trailer character used to create unique transaction names for the same SYSID running under different ADFIDs. The default is blank, and indicates nothing is specified.

If the same SYSID (for example, SAMP) runs under different ADFIDs (for example, MFC1 - ADF2), unique transaction names must be created, and scheduling must be in different message regions.

For discussions of this parameter's use in an environment with multiple IMSADF II systems, see “DEFADF Parameters” on page 80.

The Rule Generator and IMSADF II modules use the trailer to create the sign-on screen MOD name and the conversational/nonconversational APPLCTN and TRANSACT names. Figure 8 shows names created when TRXTRLR is specified and not specified for IMS.

TRXTRLR	SIGN-ON MOD NAME	CONVERSATIONAL APPL/TRAN	PSB	NONCONVERSATIONAL APPL/TRAN	PSB
=A	ssssA	ssssTtxA	ssssTtxA	ssssTAtx	ssssTAtx
None	ssss	ssssTtx	ssssTtx	ssssT0tx	ssssT0tx

Figure 8. Effect of TRXTRLR on IMS Naming Conventions

If TRXTRLR is not specified, the naming convention is not altered. When TRXTRLR is specified, the character used:

- Becomes the 5th character of the sign-on MOD name
- Becomes the 8th character of the conversational APPL/TRAN and PSB name
- Replaces the zero as the 6th character of the nonconversational APPL/TRAN and PSB name

If TRXTRLR is specified for an ADFID, all conversational and non-conversational transactions running under that ADFID use the same character, and the sign-on MOD name becomes five characters instead of four.

USRLANG The primary language in which system screens and messages are to appear. See Chapter 9, “National Language Support” for a complete discussion of use of national languages and instructions for installing one.

Enter a one-position value representing the desired language; the supplied values and their definitions are:

E English (this is the default; stored internally as blank)
F French
G German
K Korean (requires DBCS)
J Japanese (requires DBCS)
P Portuguese
S Spanish
W Swedish

This parameter can take two forms (illustrated below): either the single character, or the single character followed by a comma and the literal DBCS, all enclosed in parentheses.

USRLANG=E
USRLANG=(J,DBCS)

DBCS notation is used by some Far Eastern languages when the character set exceeds 256. DBCS is the literal you use to indicate that two bytes are required internally to represent one byte of data on the screen.

User Languages are available for you to define your own language, if one you need is not one of those listed above. Enter a numeric value, from 0-9, for a user language you define and install. (Alphabetic characters should be reserved for any future IBM-supplied languages.)

You can install the supplied screens and messages in more than one national language.

See also the ALTLANG parameter.

V50RES Indicates whether the IMSADF II online load modules MFC1V20 and ????V50 are to remain in main storage when execution is completed.

Possible values are:

V50RES=Y or YES
V50RES=N or NO

If `V50RES=YES`, ????V50 and MFC1V20 occupy space in every message region that processes IMSADF II transactions until the IMS message region is stopped. `V50RES=YES` is the recommended choice. For best performance, use this specification in conjunction with RULESBL, COMPMBL, and RULESVS.

Note: Since MFC1V20 is reentrant, it can be accessed from LPA and, therefore, need not occupy message region space.

See “IMS Storage Requirements” on page 5 for storage requirements relative to this option.

If `V50RES=NO`, ????V50 and MFC1V20 are deleted every time IMSADF II processing ends.

Note: If multiple IMSADF II transactions are in the queue, these modules remain until the QC status code is returned.

See also the RULESBL, COMPMBL, and RULESVS parameters.

WRKAREA The size of a SPA work area.

The default size is 32,767. If an IMS SPA is chosen, it cannot exceed the maximum size for a SPA (currently 32,767). Otherwise, the maximum size is 64,800.

This is an internal IMSADF II work area used to contain:

- The SPA or Work database during IMS conversational processing
- A work area during nonconversational or batch processing

WRKAREA must be large enough to cover the largest of the preceding requirements.

If you use the IMS SPA-only system or DB2, the WRKAREA value is the maximum size for your entire IMSADF II system.

If you use the IMS DL/I Work database, then:

```
WRKAREA = WRKSIZE * WRKSEGS
```

You cannot specify a value greater than 64,800 or less than 6,000 (the minimum required by IMSADF II for its supplied transactions).

If you use nonconversational processing only, specify the largest value you will need. WRKAREA has the same content as for conversational processing, although it is not carried across program iterations in nonconversational or batch processing.

See also the WRKREDU, WRKSIZE, WRKSEGS, and COMMLEN parameters.

WRKREDU Specifies whether Work database reduction (compression) is implemented.

Possible values are:

WRKREDU=N: (the default)
WRKREDU=Y

`N` indicates that Work database reduction is not to take place. The complete workarea is written out to the Work database.

`Y` indicates that Work database reduction is to take place. Before writing the WRKAREA to the Work database, IMSADF II reformats WRKAREA to eliminate Segment Layout and Input Transaction Rules' static data. Other areas are left intact. This may (based on transaction contents) reduce the number of segments written. If at least one segment is eliminated, the compressed work area is written; otherwise, the Work database is written intact. When the Work database is later read in, the WRKAREA is restored to its original condition and the required rules are reloaded. Fewer I/Os to the IMS log occur, but there is an increase in main storage requirements and execution time. There are performance considerations when you choose this option; see “Reduction/Compression” on page 95.

Note: This option is part of IMSADF II and is **not** an IMS Edit Compression routine.

This option applies to an IMS DL/I (`DB2DB=N`) environment only.

See also the WRKSEGS, WRKSIZE, and WRKAREA parameters.

WRKSEGS The maximum number of levels (segments) in the SPA Work database. WRKSEGS must be in the range of 0 to 15, where 0 means that the SPA Work database is not used by transactions running under this ADFID (all conversational data is kept in the SPA maintained by IMS storage).

WRKSEGS=0 Is the SPA-only

WRKSEGS=5 Is the supplied IMS default, since most installations use the Work database

Make sure that:

WRKSIZE * WRKSEGS ≤ WRKAREA

If you use the DB2 Work database, WRKSEGS is either 1 or 2, since there are no more than two rows in the table.

See also the WRKSIZE, WRKAREA, and SPATS parameters.

WRKSIZE The length of a SPA Work database segment (all segments are the same size). Segments must be at least 2,100 bytes long. This parameter need not be specified and is ignored if `WRKSEGS=0`.

Make sure that:

WRKSIZE * WRKSEGS ≤ WRKAREA

If you use the DB2 Work database, WRKSIZE is ignored.

See also the WRKSEGS and WRKAREA parameters.

XAMANAGE Specifies whether rules loaded above the 16 MB line are to be deleted. Possible values are:

XAMANAGE=N (the default)
XAMANAGE=Y

N indicates that rules loaded above the 16 MB line are not to be deleted until the IMS region is brought down. For performance reasons, `N` is recommended in a production environment.

Y indicates that rules loaded above the 16 MB line are to be deleted when the transaction ends. Y is recommended in a test environment because new copies of ADF rules are brought into storage each time a transaction is tested. This causes performance degredation.

ADFMFS Macro (Defining Screen Characteristics)

The basic MFS source statements used by an IMSADF II system are produced by invoking the ADFMFS macro in installation job SCREENS. Figure 9 shows the ADFMFS macro.

```
Label       ADFMFS      ADFID=MFC1,
                        MFSTRLR=1,
                        USRLANG=E,
                        SYSID=,
                        LISTING=NOGEN,
                        PAGE=DEFN,
                        DEVTYPE=(2),
                        DEVNAME=(2)
            END
```

Figure 9. ADFMFS Macro

Label Can be specified, but is ignored.

ADFID Same as ADFID for the DEFADF macro.

DEVNAME A list of the symbolic device names for which IMSADF II base MFS source statements are produced. This is the same as the DEVNAME specified in the DEFADF macro.

DEVTYPE A list of the device types for which IMSADF II base MFS source statements are produced. This is the same as the DEVTYPE specified in the DEFADF macro.

MFSTRLR Same as MFSTRLR for the DEFADF macro.

USRLANG Same as USRLANG for the DEFADF macro with an exception for National Language Support. The DBCS literal does not apply to this keyword.

If you need base screens generated for your alternate (as well as your primary) languages, run this job once for each language, changing the language code each time. Chapter 9, “National Language Support,” describes this procedure.

See also the SYSID parameter.

SYSID Used only if you support multiple languages. This keyword is the same as the second parameter of the pair you specified in ALTLANG in the DEFADF macro. This keyword corresponds to the value you enter in USRLANG. When you enter this value, your base screens have the screen formats named with the first two characters of the SYSID.

If you want base screens generated for your alternate (as well as your primary) language, run this job once for each language, changing the language code each time. Chapter 9, “National Language Support,” describes this procedure.

LISTING A specification for the MFS preprocessor PRINT statement, which controls the output listing of the MFS utility execution.

PAGE The DEV statement PAGE option used on IMSADF II printer formats.

Note: The values specified for DEVTYPE and DEVNAME in the ADFMFS macro should correspond to the value specified on the DEFADF statement. DEVNAME and DEVTYPE operands are entered as a pair with a one-to-one correspondence between the two operands. Multiple operands can be specified within each keyword. Since there is no extended attribute support for the base screens, there is no need for DEVCHRS, as all are generated with `FEAT=IGNORE`.

For example:

```
DEVTYPE=(2,2,4,3),
DEVNAME=(2,A2,A4,A03).
```

There are four terminal types defined in the IMS stage I system definition. Each terminal specified in DEVNAME has a corresponding DEVTYPE, that is, a coded description of its screen characteristics:

- Terminals 3277-2 and 3270-A2 have a screen size of 24 lines by 80 columns.
- Terminal 3270-A4 has a screen size of 43 lines by 80 columns.
- Terminal 3270-A03 is a 3279 Model 3B and has a screen size of 32 lines by 80 columns.

Output of the ADFMFS macro is created as a member in IMSADF.SCREENS, which contains source for IMSADF II base screens. The base screens are:

- Primary Option Menu
- Secondary Option Menu
- Secondary Key Selection
- Project Message Sending
- Project Message Display
- User Message Sending
- User Message Display
- Error Message
- Printer Format

The naming conventions for these screens are found in Appendix G, “MFS Naming Conventions for Supplied Screens.”

Chapter 4. Building the Customization Jobs After Initial Installation

This chapter describes using the ISPF Installation Dialogs to complete the initial installion of IMSADF II.

Invoking the ISPF Installation Dialog

The ISPF Installation Dialog is a CLIST. It is invoked by entering one of the commands below, either on the ISPF option 6 panel or in native TSO. You are prompted for ADFNODE and NEWADF, the data set high-level qualifiers allocated during installation.

```
EX(ECUTE) 'IMSADF.INSTALL(ADFSPFI)'
```

Note: The execution data set IMSADF.INSTALL was created when the product's installation JCL and procedures were unloaded from the install tape.

Alternatively, if this CLIST is copied into your CLIST.CLIST data set, then it can be invoked as follows:

```
ADFSPFI
```

You will be prompted for ADFNODE and NEWADF, which are the data set high-level qualifiers allocated during product installation.

Installation Menu

The ISPF Installation Dialogs are used to complete the installation of IMSADF II. The dialogs present a series of screens with default information already entered; you enter only those values to be changed. Validation occurs interactively. If errors are detected, you receive a message describing the error. You must correct errors before progressing to the next panel. The dialogs customize your target libraries, from which you submit the rest of the IMSADF II installation jobs, as described in Chapter 5, “Running the Customized Installation Jobs” on page 44.

Help screens (PF1) describe each parameter in detail, along with the current defaults.

Figure 10 appears after ADFSPFI is entered.

```
ADFINS0 ---------------- IMSADF II ONLINE INSTALLATION ---------------
OPTION ===>

   ADFID ===> MFC1                                    USERID   - TSOID
                                                      TIME     - 14:07
                                                      DATE     - 97/05/04
   1  INSTALL    - Install IMSADF II
   2  CHANGE     - Change Installation Options
   3  DELETE     - Delete a Version of IMSADF II
   4  CUSTOMIZE  - Customize IMSADF II libraries
   5  TUTORIAL   - Information about IMSADF II Installation
   X  EXIT       - Terminate the Installation


PRESS END KEY TO TERMINATE ONLINE INSTALLATION
```

Figure 10. INSTALL: Installation Menu

Enter the four-character ADFID with which you want to work (you can maintain multiple sets of installation characteristics). This menu provides the following options:

1 INSTALL - Do a full installation of IMSADF II.

This option presents a series of screens with detailed information about each option significant for proper installation of IMSADF II.

You choose the options you want from the menus displayed. As you complete each screen, the values you enter are saved. After the last screen is complete, the data set tailoring process begins, using your values to customize the IMSADF II target source libraries and installation job streams. You then begin submitting those jobs based on the instructions in Chapter 5, “Running the Customized Installation Jobs.”

2 CHANGE - Change installation options (described in Chapter 6, “Customizing Existing IMSADF II Systems” on page 60).

This option presents the same options as INSTALL, and you can change those options that do not require another ADFID.

3 DELETE - Delete an installed ADFID (described in Chapter 6, “Customizing Existing IMSADF II Systems” on page 60).

This option lets you delete an installed ADFID. You may have entered an ADFID erroneously or for test purposes, and now want to reclaim the space in the ISPF table in which the ADFID and its values are kept.

4 CUSTOMIZE - Customize libraries (described in Chapter 6, “Customizing Existing IMSADF II Systems” on page 60).

This option lets you customize any target libraries (or their members) used by IMSADF II.

5 TUTORIAL - Take the tutorial.

This book does not show the tutorial since it is designed to be displayed to you interactively.

X EXIT - End the installation.

Enter `X` to terminate the installation screens.

The libraries that get customized during the ISPF dialogs are:

- IMSADF.JCLLIB
- IMSADF.DBDSRC
- IMSADF.PSBSRC
- IMSADF.RULES.SOURCE

As each screen is presented, enter desired values and press ENTER to save them. Since a series of screens is typically presented, you are given the opportunity to exit at any time without saving results by entering `CANCEL`.

The examples that follow are based on detailed interactions between an IMSADF II user and the ISPF Dialog Manager.

INSTALL Option

Select option 1, as highlighted in Figure 11. Option 1 leads to six data entry screens, which contain all possible IMSADF II installation options.

These screens are scrollable, meaning that UP and DOWN keys are active and can be used to go back and forth in this sequence of screens.

Screens are presented with the defaults. Enter only the values you want changed. If an error occurs, you receive an error message. You must correct the error before leaving the panel. When you press ENTER, the validated information is saved.

```
ADFINS0 ---------------- IMSADF II ONLINE INSTALLATION ---------------
OPTION ===> 1

   ADFID ===> MFC1                                    USERID   - TSOID
                                                      TIME     - 14:07
                                                      DATE     - 97/05/04
   1  INSTALL    - Install IMSADF II
   2  CHANGE     - Change Installation Options
   3  DELETE     - Delete a Version of IMSADF II
   4  CUSTOMIZE  - Customize IMSADF II libraries
   5  TUTORIAL   - Information about IMSADF II Installation
   X  EXIT       - Terminate the Installation



PRESS END KEY TO TERMINATE ONLINE INSTALLATION
```

Figure 11. INSTALL: Installation Menu

Figure 12 is the first panel to appear. It contains general information that directs the rest of the installation process.

```
ADFINS1 --------------- GENERAL OPTIONS ---------------  PAGE 1 OF 5
COMMAND ===>

 ADFID                  ==> MFC1

 Environment            ==> I       (I=IMS)
 ADF Databases in DB2   ==> N       (Y=Use DB2, N=Use DL/I)
 Transaction Trailer    ==>
 V50 Resident ?         ==> Y       (Y/N)
 Format of Dates        ==> U       (U,S,B,E,O)
 Assembler Name         ==> F       (A=A5MA90, H=ASSEMBLER H)
 Key Audit Names        ==> S       (S=standard,A=alternate)
 User Exit/SPR Format:      L       (L=load format)
 IADF Feature ?         ==> Y       (Y/N)
 DB2 Extension ?        ==> N       (Y/N)
 Data Dictionary ?      ==> N       (Y/N)
 Programming Language   ==> C       (C=COBOL,P=PLI,A=ASSEMBLER)
 RACF Support           ==> T       (U=RACFUSER,T=TERMINAL,R=RULEGEN)
 End of text Delimiters ==> $$      (Used to signal end of text data)
 Primary Language       ==> E       (E,F,G,J,K,P,S,W or 0-9)
 Alternate Languages ?  ==>         (Y/N)

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 12. INSTALL: General Options

The most important entries on this panel are:

- **ADFID**
- **Environment**
- **ADF Databases in DB2**
- **Transaction Trailer**

Once you have installed this ADFID, you cannot change these four values, as they all require installation of another ADFID. See Chapter 7, “Multiple IMSADF II Systems” on page 79 for a discussion of those implications.

Based on your database access method for the IMSADF II dynamic rules databases (DB2DB), some values on the following panels are protected. That is, you are not able to enter information in some of the fields. This is because certain entries don't apply in the environment you have chosen.

Note: The distributed default subsystem is IMS, with DL/I access used for the dynamic rules databases. The panels shown in this chapter correspond to that choice. Your panels will look different if you select other values.

The field **DB2 Extension?** is not the same as **ADF Data Bases in DB2?**. **DB2 Extension?** selects the RGLGEN utility for inclusion in the IMSADF II installation link-edits, as well as selecting other DB2 information pertinent to library customization. You can use DB2 tables for your own applications, while retaining DL/I access for the IMSADF II dynamic rules databases (`DB2DB=N`).

You must respond to the **Alternate Languages?** question, which normally pertains to using multiple foreign languages in IMSADF II (see Chapter 9, “National Language Support”).

If you enter `Y` beside **Alternate Languages**, Figure 13 appears (otherwise you see Figure 14).

```
ADFINS1A --------------- ALTERNATE LANGUAGES ------------- PAGE 1A
COMMAND ===>

 ADF SYSTEM ID: MFC1

           Language      SYSID     DBCS (Y/N)

                                     N
                                     N
                                     N
                                     N
                                     N
                                     N
                                     N
                                     N
                                     N
                                     N

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 13. INSTALL: Alternate Languages (NLS support)

Enter the letter value for the alternate language(s) you select, a four-character SYSID for base screens and messages in that language, and whether DBCS (Double Byte Character Set) notation applies.

If you enter `N` beside **Alternate Languages**, you see Figure 14.

```
ADFINS2 ----- INSTALLATION SCREEN CHARACTERISTICS -----   PAGE 2 OF 5
COMMAND ===>

 ADF SYSTEM ID: MFC1


 Device Name(s)            ==> 2
 Device Type(s)            ==> 2
 Device Characteristics    ==> 0
 No. of Default Terminals  ==> 1         (1-5)
 MFS Trailer               ==> 1
 Default Printer LTERM     ==> PRINTER
 MFS PRINT statement       ==> N         (G=gen,N=nogen)
 DEV statement PAGE option ==> D         (F=float,D=defn,S=space)
 Format for Literals       ==> D         (L=left,R=right,D=dash)
 MFSUTL Procedure Name     ==> MFSUTL

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 14. INSTALL: Define Screen Options

Figure 14 contains the installation's screen characteristics, terminal default characteristics, and the procedure name used to compile IMS screen formats.

After the screen options are validated (the number of device names, types, and characteristics must be the same), Figure 15 appears.

```
ADFINS3 ---- INSTALLATION DATA SET CHARACTERISTICS -------PAGE 3 OF 5
COMMAND ===>

 ADF SYSTEM ID: MFC1

 Node for Shared Files   ==> IMSADF             (ADFNODE)
 For Non-shared Files    ==> IMSADF             (NEWADF)
 Old IMSADF II Node      ==> IMSADF             (OLDADF)
 IMS/VS Node             ==> IMSVS              (IMSNODE)
 IMS/VS Node (DBD/PSB)   ==> IMSVS              (IMSTEST)
 ISPF Node               ==> SPF.PRODUCT        (For ISPLINK)
 Data Dictionary Node    ==>                    (For DDLOAD)
 DB2 Node                ==>                    (For DB2 Libraries)
 SMP Node                ==> IMSADF             (SMPNODE)
 PROCLIB Data Set Name   ==> SYS1.PROCLIB       (For IMSADF Procs)
 Rules Library DDNAME    ==>
 No. of BLDL Entries     ==> 0
 Use Composite Modules ? ==> N                  (Y=yes,N=no,A=alt)
 Composite DDNAME        ==>
 No. of BLDL Entries     ==> 0

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 15. INSTALL: Define Nodes

On this screen, you enter the high-level qualifiers for IMSADF II libraries (the qualifiers must be the same as for installation job ADFAALOC) and the ddnames (if desired) of the rules and composite libraries.

Based on the subsystem (ENVIR) and database access method for the IMSADF II dynamic rules databases (DB2DB) you selected in Figure 12, you cannot enter information in certain fields; this is because they do not apply to your environment.

The next set of installation options is shown in Figure 16. The figure contains specific subsystem parameters and procedure names.

```
ADFINSA ----- STORAGE INFORMATION AND MISCELLANEOUS  --- PAGE 4 OF 5
COMMAND ===>

 ADF SYSTEM ID: MFC1

 Available Storage        ==> 0       (0-9999K for Rules Management)

 DB2 Information                      If DB2 used for ADF Data Bases
 Subsystem Name           ==>
 Data Base Name for ADF   ==>
 Storage Group            ==>
 High Level Table Node    ==>         (SYSADF)

 Procedure Information
 DBDGEN Procedure Name    ==> DBDGEN
 PSBGEN Procedure Name    ==> PSBGEN
 ACBGEN Procedure Name    ==> ACBGEN
 Esoteric Name (DASD)     ==> SYSDA   (for Temporary Space)

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 16. INSTALL: Storage and Subsystem Information

Based on the subsystem (ENVIR) and database access method for the IMSADF II dynamic rules databases (DB2DB) you selected in Figure 12, you cannot enter information in certain fields; this is because they do not apply to your environment.

If you selected DB2 access for the IMSADF II dynamic rules databases, you complete the middle section of this panel, indicating generally the placement and definition of the IMSADF II tables you want to create.

Enter any special procedure names for DBDGEN, PSBGEN, and ACBGEN processes.

The last set of installation options is presented in Figure 17.

```
ADFINS4 - INSTALLATION WORK DATA BASE/SMP INFORMATION -- PAGE 5 OF 5
COMMAND ===>

 ADF SYSTEM ID: MFC1


 SPA or Work Area Size    ==> 30000      (From 6000 to 64800) WRKAREA
 Use Work Data Base ?     ==> Y          (Y/N)
 Maximum no. of Segments  ==> 5          (From 1-15)          WRKSEGS
 Segment Size             ==> 6000       (At least 2100)      WRKSIZE
 Work Area Reduction ?    ==> N          (Y/N)
 Communication Area Size  ==> 0          (Used at Sign-on Time)

SMP INFORMATION
 Version                  ==> E          (4 or E)

 SMP/E only:
 GLOBALZONE Data Set Name ==>
 DLIBZONE name            ==>            (DDNAME)
 TARGETZONE name          ==>            (DDNAME)

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 17. INSTALL: SPA and SMP Options

This panel contains SPA and SMP information. Here the process validates size restrictions and reports possible errors to you.

The important values to be entered on this panel are **WRKSIZE**, **WRKSEGS**, and **WRKAREA**. Their relationship is described in “DEFADF Macro (Defining Basic IMSADF II Characteristics)” on page 15. The values you must enter depend on the environment you selected in Figure 12:

- In IMS DB2, the Work database table is defined as LONG VARCHAR, so **WRKAREA** is the only value to enter.
- In IMS DL/I, all values can be entered, and the IMS SPA is chosen if **WRKSEGS=**0.

After all data collection is complete, Figure 18 appears, representing the last step in online installation.

See IMSADF.INSTALL (ADFINTE) for the following values:

- GLOBALZONE Data Set Name
- DLIBZONE name
- TARGETZONE name

```
ADFINS7 ------------- CUSTOMIZE IMSADF II SOURCE LIBRARIES ----------
COMMAND ===>

                                                    USERID   - TSOID
                                                    TIME     - 14:09

Customize all IMSADF II        libraries


       USERMOD name ===>



JOB STATEMENT INFORMATION:      (No JOBS will be submitted)
  ===> //TSOID    JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*
  ===> //*

Press Enter to Customize ALL libraries or END, CANCEL to exit
```

Figure 18. INSTALL: Customization Process

This panel lets you customize selected members of IMSADF II ADFSLIB, based on the values you entered, and store those members in the following four target libraries:

- IMSADF.JCLLIB
- IMSADF.DBDSRC
- IMSADF.PSBSRC
- IMSADF.RULES.SOURCE

The contents of these libraries vary based on the options you entered on the panels.

Note: Compressing the libraries before trying more than two full customizations avoids an out-of-space condition.

Enter the seven-character name of the SMP USERMOD you want to create.

Specify the job JCL statements that will be used in **every** customization job that is created (no syntax checking is done on these statements).

If you exit from this screen before customization is done, your installation values are **not** saved. The customization process is lengthy, since four libraries are involved. Once customization is complete, Figure 19 is presented.

```
---- CUSTOMIZE IMSADF II SOURCE LIBRARIES --- All members customized
COMMAND ===>
                                                        USERID   - TSOID
                                                        TIME     - 14:09

Customize all IMSADF II        libraries


       USERMOD name ===> ADF0001



JOB STATEMENT INFORMATION:      (No JOBS will be submitted)
  ===> //TSOID    JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*
  ===> //*
```

Figure 19. INSTALL: Completed Customization

The INSTALL option is now finished.

In addition to the customized libraries, another output of INSTALL is the addition of a row to ISPF table IMSADF.ADFTLIB(PRODUCT), which contains the options you have just selected.

You should now go to Chapter 5, “Running the Customized Installation Jobs” for instructions on completing installation of IMSADF II.

Chapter 5. Running the Customized Installation Jobs

This chapter summarizes the tasks involved in the final phase of installing IMSADF II.

Before doing this task, you must successfully complete the ISPF Installation Dialogs INSTALL option (Chapter 4, “Building the Customization Jobs After Initial Installation”), which results in the creation of IMSADF.JCLLIB.

This chapter lists the jobs associated with each installation task. The jobs you must execute are based on your subsystem and database access for the IMSADF II dynamic rules databases (DEFADF parameters DB2DB and ENVIR). This chapter has five major sections:

Do the first **or** second section, as appropriate for your environment. Then do the third section. The fourth section will help you if errors occur. The fifth section contains descriptions of the jobs in IMSADF.JCLLIB and any special information you may find useful (such as return codes) for each job. The job descriptions are in alphabetical order.

When you have completed all the jobs or tasks in this chapter, you have completed installation of IMSADF II for the features and environment that apply at your installation.

Installation Jobs for the IMS DL/I Environment

Figure 20 lists the jobs you must successfully complete for this environment.

Figure 20. IMSADF II Installation Steps for IMS DL/I Environment

Step	Job Name For This Step	Description
1	ADFOPT	Install the DEFADF options you specified during the ISPF Installation Dialogs.
2	JCLNLINK	Create all SMP LMOD entries for job ADFLINK.
3	COPYPROC	Copy IMSADF II procedures to the PROCLIB of your choice, for example, SYS1.PROCLIB.
4	ADFLINK	Link-edit IMSADF II modules, Rule Generator for system, and sample problem.
5	ADFDBD	Do DBDGENs for IMSADF II and Work database, if used.
6	ADFPSB	Do PSBGENs for IMSADF II system transactions and sample problem.
7	ADFACB	Do ACBGENs for IMSADF II system transactions and sample problem.
8	ADFISPU	Link IADF feature modules with ISPLINK. **Do this step only** if you selected this feature.
9	SCREENS	Customize unique system screens.
10	MFSJOB	Run MFS utility for IMSADF II screens for system transactions and sample problem.
11	ADFWORKI	Allocate and initialize Work database, if you use one. **Execute this step only** if you selected the Work database in ADFOPT.
12	ADFDATA	Load IMSADF II dynamic rules databases with supplied data. **Execute this step only** if you are a new IMSADF II user.
12	DBUPDATE	Update IMSADF II dynamic rules databases with supplied data. **Execute this step only** if you are a current IMSADF II user.
13	ADFSTG1	Use these statements to help you do an IMS generation.

See “Additional Tasks to Complete” on page 47 for additional tasks that may apply in your environment.

Note: While **it is not recommended** that IMSADF II releases be skipped, Appendix H, “Currency Requirements for Skipped Releases” contains guidelines for installing Version 2 Release 2 correctly. The procedures found there **replace** job DBUPDATE for existing users who do not have IMSADF II Version 2 Release 1 installed.

Installation Jobs for the IMS DB2 Environment

Figure 21 lists the jobs you must successfully complete for this environment.

Figure 21. IMSADF II Installation Steps for IMS DB2 Environment

Step	Job Name For This Step	Description
1	ADFOPT	Install the DEFADF options you specified during the ISPF Installation Dialogs.
2	JCLNLINK	Create all SMP LMOD entries for job ADFLINK.
3	COPYPROC	Copy IMSADF II procedures to the PROCLIB of your choice, for example, SYS1.PROCLIB.
4	ADFLINK	Link-edit IMSADF II modules, Rule Generator for system, and sample problem.
5	ADFPSB	Do PSBGENs for IMSADF II system transactions and sample problem.
6	ADFACB	Do ACBGENs for IMSADF II system transactions and sample problem.
7	ADFISPU	Link IADF feature modules with ISPLINK. **Execute this step only** if you selected this feature.
8	SCREENS	Customize unique system screens.
9	MFSJOB	Run MFS utility for IMSADF II screens for system transactions and sample problem.
10	ADF2PAL	Precompile, assemble, and link the Master and Work database Table Handler rules.
11	ADFTBTSO	Run all steps to create tables, bind plans, and grant authority, using batch TSO with DB2 attach.
12	ADFTRTR	Translate batch driver input from DL/I to DB2 format.
13	DBUPDATE	Update IMSADF II dynamic rules databases with supplied data. **Execute this step only** if you are a current IMSADF II user.
14	ADFSTG1	Use these statements to help you do an IMS generation.

See “Additional Tasks to Complete” on page 47 for additional tasks that may apply in your environment.

Additional Tasks to Complete

Do any of the following tasks that apply at your installation.

Updating JCL for the Online Control Region

At this point, update the JCL used to start the online IMS control region. Before executing IMSADF II online with IMS, the JCL used to start the IMS control region can be updated with DD statements for IMSADF II dynamic rules databases and, if it is used, the Work database. The JCL is in “ADFWORKI (Initialize the DL/I Work Database)” on page 56.

Modify all JCL statements that include IMSADF II database information. JCL statements can be found in Appendix A, “Sample IMSGEN Control Statements” on page 150 or in Appendix B, “Sample Problem Examples” on page 152. Be sure to include DD statements for the IMS-supplied sample problem database.

If you use the DL/I Work database, make sure your value for the PSB work area (PSBW) is large enough.

Executing an Online Installation Verification Procedure

In the sample problem, data is added to the IMS sample database, manipulated, and then deleted. The sample problem verifies that the dynamic rules databases have been correctly loaded and that IMSADF II functions operate correctly. The sample problem is available to DL/I users. The online installation verification procedure is described in detail in Chapter 11, “Sample Problem.”

Tasks for Current IMSADF II Users Only

These tasks do **not** apply to new users.

Regenerating Transaction Drivers

If you have developed applications using previous releases of IMSADF II, you must invoke the Rule Generator to relink all your transaction drivers. The restructuring of IMSADF II programs to achieve reentrancy changed the driver modules.

Preparing TSO and ISPF for IADF

Before IADF is ready to use, certain administrative tasks must be complete. The IADF administrator must prepare TSO and ISPF. The ISPF tables that IADF requires for each ADFID and SYSID for which development is desired must be initialized.

See *IMS Application Development Facility II Version 2 Release 2 Interactive ADF Administration Guide* for more information.

Installing a Lockword Exit

Your installation may have a Lockword exit. See “Installing Sign-On and Sign-Off Exits” on page 77 for instructions on installing a Lockword exit at this time.

Preparing IMSADF II to Use DB2

A program linked in job ADFLINK is known as RGLGEN. You do not need the RGLGEN to develop an IMSADF II transaction that can access DB2 tables. RGLGEN extracts Rule Generator source from the DB2 catalog, insuring that your input is synchronized with the DB2 definition. See *IMS Application Development Facility II Version 2 Release 2 DATABASE 2 Application Specification Guide* for further instructions.

Preparing IMSADF II to Use Data Dictionary Model

See *IMS Application Development Facility II Version 2 Release 2 Data Dictionary Extension User's Guide* for further instructions.

Installing an Alternate or User-Defined Language

If you install either an alternate language or a user-defined language, see Chapter 9, “National Language Support” for installation instructions.

Installing DB2 Support Modules

A DB2 support program is known as RGLGEN. It is not necessary for you to have the RGLGEN to develop an IMSADF II transaction that can access DB2 tables. RGLGEN extracts Rule Generator source from the DB2 catalog, thus insuring that your input is synchronized with the DB2 definition.

Note: This option is not the same as being able to put the IMSADF II dynamic rules databases in DB2 table format. The parameter that does this is DB2DB, and you cannot change its value after initial installation.

1. Make sure all maintenance is in ACCEPT status, including the usermod for ????SYSP (otherwise, you do not get your current installation options).
2. Add the DB2 DLIB ALOAD to your SMP procedure (or DDDEF).
3. Select the CHANGE option on the main Installation Menu and follow it through to completion, indicating you want the DB2 extension. This changes your installation options in the PRODUCT table in IMSADF.ADFTLIB.

   Since this option is not stored in ????SYSP, you do **not** have to run job ADFUSER.
4. CUSTOMIZE the JCLLIB member ADFLINK.
5. Run step L200 from job ADFLINK, which links the IMSADF II DB2 support modules.
6. Run job JCLNLINK, which does JCLIN processing against job ADFLINK.
7. See *IMS Application Development Facility II Version 2 Release 2 DATABASE 2 Application Specification Guide* for information on developing IMSADF II applications using DB2.

Installing the Data Dictionary Extension

1. Make sure all maintenance is in ACCEPT status, including the usermod for ????SYSP (otherwise, you do not get your current installation options).
2. Add the Data Dictionary DLIB DDLOAD to your SMP procedure.
3. Select the CHANGE option on the main Installation Menu and follow it through to completion, indicating you want the Data Dictionary extension.

Since there are no Data Dictionary options stored in ????SYSP, you do **not** have to run job ADFUSER.

4. CUSTOMIZE the JCLLIB members ADFLINK, DDEECI, DDEECI5, and DDESAMP.
5. You can then run step L100 from job ADFLINK, which links the IMSADF II Dictionary support modules.
6. Run job JCLNLINK, which does JCLIN processing against job ADFLINK.
7. See *IMS Application Development Facility II Version 2 Release 2 Data Dictionary Extension User's Guide* for information on establishing your model.

Installing an Alternate or User Language

This option is of interest primarily to non-English speaking installations. See the chapter on NLS support.

Errors During INSTALL

The situation may arise in which you have completed INSTALL, have customized libraries, and then realize while reviewing job ADFOPT that you specified an option incorrectly. The following discussion applies ONLY when:

- You have completed the INSTALL option of the ISPF Installation Dialogs.
- You have not yet run job ADFOPT.

The PRODUCT table is already updated, so the task is to make the DEFADF options match those in PRODUCT. You can do this in two ways:

Method 1. Select the DELETE option for the ADFID; then INSTALL the ADFID again correctly.

This approach is the most straightforward and is preferred for that reason. The disadvantage is that all libraries are recustomized. If, however, your change is such that IMSADF.JCLLIB requires customization (for example, you specified an incorrect IMSNODE), this approach is proper. See Figure 40 for a table of dialog options requiring some recustomization. “Changes Requiring No Additional Steps” on page 74 lists the options requiring no customization.

Method 2. The following method does not require that all libraries be recustomized, but alters slightly the jobs you submit in Chapter 5, “Running the Customized Installation Jobs”:

1. Run job ADFOPT to completion, which performs an SMP ACCEPT.
2. Select CHANGE and make the corrections. The PRODUCT table is now correct, but your ????SYSP module is not.
3. Run job ADFUSER to completion, ACCEPTing it.

   This puts two usermods on ????SYSP. Because you have not linked the IMSADF II modules yet, you must have ACCEPTed the usermod, so the correct options are incorporated when you run job ADFLINK later.
4. CUSTOMIZE any required library or member indicated by the change. See Figure 40 on page 73 for a table of dialog options requiring some recustomization.

5. Continue with installation, beginning with the job following ADFOPT in your particular environment (as described in Chapter 5, “Running the Customized Installation Jobs”).

Contents of IMSADF.JCLLIB

The jobs described in this section were customized during the ISPF Installation Dialogs. All jobs reside in IMSADF.JCLLIB. Jobs are listed in alphabetical order.

The contents of IMSADF.JCLLIB differ based on whether you are using:

- IMS subsystem and DL/I dynamic rules databases support
- IMS subsystem and DB2 dynamic rules databases support

ADFACB (Execute ACBGENs for Supplied Transactions)

This job creates application control blocks (ACBs) from IMSADF II DBDs and PSBs. The ACBs are put into the IMSVS.ACBLIB library.

This job generates ACBs by executing the IMS ACBGEN procedure. Input to the procedure is contained in the JCL.

Notes:

1. Make sure that the DBD for DI21PART (distributed with IMS) is in the //IMS DD concatenation. (Otherwise, the ACBs for SAMPVCD and SAMPTOR are not built.)
2. If the DBDs do not already exist in your ACBLIB, you receive a return code of 08 from this job; this is normal for first time users.

ADFDATA (Load IMSADF II DL/I Dynamic Rules Databases)

ADFDATA initially loads the IMSADF II DL/I dynamic rules databases.

This job loads the IMSADF II Audit, Message, and Sign-On Profile databases by executing the IMSADF II batch transaction driver. The dynamic rules databases were allocated and cataloged in job ADFAALOC. The input to this job is a member called TRANSIN in IMSADF.JCLLIB. The key (X'FF') supplied for each database forces initialization of the entire root addressable area (RAA) using the DFSHDC40 randomizer.

ADFDBD (Execute DBDGENs for IMSADF II DL/I Databases)

This job generates all IMSADF II DBDs by executing a series of IMS DBDGEN procedures. Members in IMSADF.DBDSRC are input to these procedures.

Note: These DBDs should be reviewed before this job is run. The DBDs specify DFSHDC40 (supplied as part of the IMS system) as the randomizing module for IMSADF II dynamic rules databases.

This job uses the DBDGEN utility to generate DBDs for IMSADF II dynamic rules databases and, if it is used, the IMSADF II Work database.

You are encouraged to examine the supplied DBDs and their parameters in light of your installation's requirements. Since the root segment key is the terminal (LTERM) name, you can substitute your own randomizer. A sequential randomizer

gives the best Work database performance with respect to PI waits and deadlocks. You can also change the access method. For additional information on the substitutions you can make, see the IMS library. “IMSADF II Dynamic Rules Databases Considerations (IMS and DB2)” on page 7 provides more information about distributed DBDs.

ADFISPU (Link IADF Feature with ISPLINK)

This job is created only if you indicate that the IADF feature is to be installed. No matter how many ADFIDs you install in a release of IMSADF II, you only have to install the IADF feature once.

This job "adds" ISPLINK to the IADF feature as a usermod (whose name you can change); it puts ISPLINK into a temporary data set during the ACCEPT from the real ISPF load library found in the //ISPLOAD DD statement.

Change the data set name of the //ISPLOAD ddname to your installation's standard, as needed.

The completion code should be 0 from SMP RECEIVE, APPLY, and ACCEPT.

ADFLINK (Link-Edit and Generate Rules for Supplied Transactions)

This job link-edits the execution time IMSADF II load modules based on the ADFID specified in ADFOPT. This job must be run to link-edit IMSADF II common modules and to generate the rules for the IMSADF II system being installed. The target libraries for the resulting load modules are IMSADF.ADFLOAD and IMSADF.RULLIB. The normal return code is zero for all steps, except where noted otherwise.

The expected return codes from the steps are as follows:

L010	4
L020	4
L030	4
L040	4
L050	4
L060	4
G1.AUDIT	0
G1.MSG	0
G1.SIGNP	0
G1.BASE	0
G1.BATCH	0
G1.BATSP	0
G1.DRULG	0
G1.BDDRG	0
G1.SAMP	0
L080	4

Notes:

1. This item does not apply if your environment is DB2.

   Step L080 (the link-edit for ????V50) has a LOAD DD statement referencing the IMSVS.RESLIB library, where the language interface DFSLI000 resides. If you share SMP data sets with IMS, this ddname and data set name must match that in your IMS SMP procedure for the DLIB for MOD(DFSLI000). Since supplied rules are included in ????V50, it is linked after generation (see Figure 12 on page 37). ????V50 references the distribution library where the TSO DB2 language interface DSNELI resides. This ddname and data set name must match that in your DB2 SMP procedure for the DLIB for that MOD(DSNELI).

2. Step L100 (the dictionary link-edit) has a DDLOAD DD statement referencing the distribution library where the language interface DBDWLNKA resides. If you share SMP data sets with IMS, this ddname and data set name must match that in your IMS SMP procedure for the DLIB for that MOD(DBDWLNKA). This step is present only if you specified the step in the ISPF Installation Dialogs (Figure 12).

3. Step L200 (the DB2 support link-edit) has an ALOAD DD statement referencing the distribution library where the TSO DB2 language interface DSNELI resides. This ddname and data set name must match that in your DB2 SMP procedure for the DLIB for that MOD(DSNELI). This step is present only if you specified it in the ISPF Installation Dialogs (Figure 12).

4. To avoid having modules become back-level:

   - All modules that are in apply status should be accepted, **or**
   - The target library should be included, along with the distribution library, before this job is run.

ADFOPT (Customize IMSADF II Installation Options)

In this job, you establish your installation's options for this release of IMSADF II. By invoking the DEFADF macro, you define unique load module names, database names, library nodes, and options.

This job places a usermod on your SMP data sets. It adds a new source member and module to your IMSADF II distribution libraries. The name you select is unique: a four-character ADFID plus SYSP. Since it is not part of the base product, this job is not affected by IBM maintenance. The usermod must be ACCEPTED before you can go further in installation; **but it is best to run the job once for RECEIVE and APPLY, verify that the results of DEFADF are correct, and then run this job again for the ACCEPT, commenting out the RECEIVE and APPLY statements.**

Notes:

1. Some `MNOTE 0` messages may appear in the assembly due to the environment you specify, but they are informational.
2. The APPLY results in a return code of 0.
3. The ACCEPT results in a return code of 4.

If you have errors in APPLY, either from SMP or from the DEFADF invocation, special steps must be taken. You cannot RESTORE the usermod, since neither

the SRC nor the MOD whose name you specified exists in the IMSADF II distribution library until you ACCEPT the usermod. You must use UCLIN processing to delete the usermod entries from the SMP data sets. See “UCLDEL (Delete ????SYSP Entries Using UCLIN DELETE)” on page 59, which is the job provided to do this. UCLIN is not a normal part of installation and is to be used only if you want to rerun ADFOPT before ACCEPTing your usermod.

ISPF Table Considerations

If job ADFOPT fails and you must use job UCLDEL, you have to synchronize PRODUCT with your ????SYSP module. This is done as follows:

1. Run UCLDEL to delete the usermod and its components.
2. DELETE the ADFID using the ISPF Installation Dialogs.
3. INSTALL, entering the correct options.
4. Run job ADFOPT to successful completion.

PRODUCT now matches the options in ????SYSP.

ADFPSB (Execute PSBGENs for IMSADF II-Supplied Transactions)

This job uses the PSBGEN utility to generate the PSBs used by IMSADF II.

This job generates all IMSADF II PSBs by executing a series of IMS PSBGEN procedures. Members in IMSADF.PSBSRC are input to the procedures.

IMSADF.ADFMAC is concatenated with SYSLIB, since IMSADF II macros are invoked.

Note: If your installation chose the IMS environment and DB2 database support, some PSBs consist of only the three TPPCBs.

ADFRELOD (Reload SMP Entries for IMS Release Change)

This job is necessary only if you installed IMSADF II on IMS SMP data sets.

This job reloads the IMSADF II entries onto SMP data sets that were previously unloaded by job ADFUNLOD. This job is normally done when an IMS release change occurs or if an ALL generation is required.

See “IMS Release Change” on page 109 for a complete discussion of maintenance considerations.

ADFSTG1 (Execute Stage I System Definition)

The final phase of installation is either a full or a nucleus-only IMS system definition.

This system definition must be done to incorporate IMSADF II into IMS. This job can be executed independently of your IMSADF II installation. ADFSTG1 provides IMS generation samples (see Appendix A, “Sample IMSGEN Control Statements” for a listing of this job).

The MSGQUEUE requires a minimum of 1728 bytes for the long message queue. Review and modify the SPAREA macro and the SPA parameter of the TRANSACT macro, as applicable. Chapter 2, “Preparing the IMSADF II Environment” provides

information on how to complete the SPA parameter. The SPAREA macro must conform to your installation's decision to use only the SPA, or a SPA and HDAM Work database combination.

The examples supplied in ADFSTG1 are for the SPA/Work database combination. Appendix D, “SPA Size Calculations” gives guidelines on how to estimate the size of a SPA work area for IMSADF II transactions.

Notes:

1. If you are using the DL/I SPA/Work database combination, you must allocate and initialize the HDAM Work database.
2. The sample problem uses the distributed IMS sample problem database. Therefore, IMS system generation should include the `DATABASE DBD=DI21PART` macro specification.

ADFTBLB (Bind IMSADF II Plans for DB2)

This member binds all plans for IMSADF II-supplied transactions and drivers.

For normal installation, this member is input to job ADFTBTSO, which processes all required DB2 functions in one job. However, you could execute ADFTBLB independently using SPUFI.

ADFTBLD (Create Database/Tablespaces for DB2)

This member creates the database using DB2. It creates the database in an existing storage group. Tablespaces are defined within the database using DB2, not IDCAMS. The database name, storage group name, and high-level tablespace nodes are customized from the values you entered on the ISPF Installation Dialogs panels.

For normal installation, this member is input to job ADFTBTSO, which processes all required DB2 functions in one job. However, you could execute ADFTBLD independently using SPUFI.

You must have DB2 authority of at least CREATDBA and the USE privilege for BP32 (if your subsystem is IMS) and BP1 to complete this step.

If you want to use IDCAMS to allocate space for your tables, modify this member.

ADFTBLDR (Drop IMSADF II Dynamic Rules Databases for DB2)

This member drops all tables, tablespaces, and the database. This member should be used when you want to rerun job ADFTBTSO, which processes all required DB2 functions in one job.

You can execute ADFTBLDR independently using SPUFI. By editing ADFTBTSO, you can concatenate it ahead of ADFTBLD in STEP1, and do the entire task in one job.

ADFTBLF (Free IMSADF II Plans for DB2)

This member frees all plans, preparing for a rebind using ADFTBLB or ADFTBTSO. (ADFTBTSO processes all required DB2 functions in one job.)

You can execute ADFTBLF independently using SPUFI.

ADFTBLG (Grant Authority for IMSADF II Plans for DB2)

TSOID, as supplied, is given authority to bind the installation-supplied plans. PUBLIC is given EXECUTE authority on all the plans except ????TCT, which is granted only to you. You may want to change the authorizations before invoking ADFTBTSO, consistent with your administration of DB2. You must have the DB2 authority to execute these commands.

For normal installation, this member is input to job ADFTBTSO, which processes all required DB2 functions in one job. However, you could execute ADFTBLG independently using SPUFI.

ADFTBLT (Create Tables for IMSADF II)

This member creates all DB2 tables that correspond to the DL/I IMSADF II dynamic rules databases. These tables are as follows:

- Audit
- Message
- Sign-on
- Work

For normal installation, this member is input to job ADFTBTSO, which processes all required DB2 functions to create the tables in one job. However, you could execute ADFTBLT independently using SPUFI.

If you want to put certain tables in their own tablespaces (providing you created them in ADFTBLD), modify this member. See Appendix E, “DB2 Space Allocation Calculations” for more detailed information.

ADFTBTSO (Prepare to Load IMSADF II Tables for DB2)

This job invokes TSO in batch, with the DB2 ATTACH facility, to do the following:

- Define dynamic rules databases and table spaces (see ADFTBLD)
- Create tables for IMSADF II dynamic rules databases (see ADFTBLT)
- Bind installation-supplied plans (see ADFTBLB)
- Grant authority to execute IMSADF II plans (see ADFTBLG)

For convenience, you can run each one of the above functions as several job steps. Alternatively, you can do each task separately through SPUFI.

ADFTRTR (Translate Batch Driver Input to DB2)

This job translates the batch driver DL/I input (TRANSIN and TRANSUP) to DB2 table input format. Both input and output are placed as members in IMSADF.JCLLIB in customized form. The naming convention replaces the characters ANS in the member input with DB2 for output member names (for example, TRANSIN becomes TRDB2IN).

You must FREE IMSADF.JCLLIB before submitting this job if the library is currently allocated due to invocation of the ISPF Installation Dialogs. IMSADF.JCLLIB is used as an output library.

ADFULNG (Add User Language)

This job is necessary only if you want to support a national language that is not supplied by IMSADF II. See Chapter 9, “National Language Support” for information on National Language Support.

This job adds the following elements, which you previously translated from the supplied English version into your desired language, to IMSADF II:

- MFC1G40n
- MFC1V36n
- TRANSINn
- TRANSUPn
- ADFMFSLn

ADFUNLOD (Unload SMP Entries for IMS Release Change)

This job is necessary only if you have installed IMSADF II on the IMS SMP data sets.

This job unloads the IMSADF II SMP entries. Normally this should be done when an IMS release change occurs or if an ALL generation is required.

See “IMS Release Change” on page 109 for a complete discussion of maintenance considerations when this occurs.

The IMSADF II SMP entries should be reloaded using job ADFRELOD.

ADFUSER (Change DEFADF Options After Installation)

This job is executed only after you have invoked the ISPF Installation Dialogs CHANGE option. The CHANGE process updates the installation PRODUCT table with your changed options. To incorporate the changes in the IMSADF II execution modules, you must RECEIVE and APPLY this usermod, which replaces your ????SYSP control block, initially added during ADFOPT processing. Some MNOTE 0 messages may appear in the assembly due to the environment you specify, but they are informational.

It is critical that you run this job whenever you change options so that the PRODUCT table and ????SYSP are synchronized. See Chapter 6, “Customizing Existing IMSADF II Systems” for a complete discussion of this topic.

ADFWORKI (Initialize the DL/I Work Database)

This job allocates direct-access storage space for the IMSADF II Work database. The key in this job consists of 8 bytes of X'FF', which forces initialization of the entire root addressable area (RAA) using the DFSHDC40 randomizer.

Change any IMS release-related parameters in the supplied JCL so that they are compatible with the release or features at your installation (for example, IMSID, IRLM, or DBRC parms).

ADF2PAL (Precompile, Assemble, Link Master Table Handler Rules)

This job uses an instream procedure similar to DSNHASM to precompile, assemble, and link the master Table Handler rules ????MSTR, ????SPAW, and ????SPRS. Master Table Handler rules access all IMSADF II dynamic rules database tables and must be bound with all your developed applications.

The precompile step (PC) completes normally with a return code of 4.

COPYPROC (Copy IMSADF II Procedures to PROCLIB)

This job copies the supplied IMSADF II cataloged procedures into the procedure library you entered on the ISPF Installation Dialogs panels; this job uses the ADFID name specified in job ADFOPT.

This job results in a condition code of 0.

Note: This job replaces existing procedures that have duplicate names.

DBUPDATE (Update IMSADF II Dynamic Rules Databases/Tables)

This job updates the IMSADF II Audit, Message, and Sign-On Profile databases by executing the IMSADF II batch transaction driver.

Note: While it is **not recommended** that IMSADF II releases be skipped, Appendix H, “Currency Requirements for Skipped Releases” contains guidelines you must follow to install Version 2 Release 2 correctly. The procedures found there **replace** this job for existing users who do not have IMSADF II Version 2 Release 1 installed.

ADFDATA initially loads the IMSADF II dynamic rules databases if DL/I is used. DBUPDATE updates the dynamic rules databases using TRANSUP as input.

When DB2 is used for those databases, DB2 initially loads them using a member of IMSADF.JCLLIB called TRDB2IN (see ADFTRTR). For update, the member name is TRDB2UP. The tables were defined in job ADFTBTSO. For DB2, DBUPDATE must run as an IMS BMP transaction.

DDEECI (Add Extensibility Categories)

This member is created only when you:

- Indicate that you use the IMSADF II Data Dictionary extension on the installation panel
- Choose DL/I database support

The functions done by this member are described in the *IMS Application Development Facility II Version 2 Release 2 Data Dictionary Extension User's Guide*.

DDEECI5 (Modify Data Dictionary Model)

This member is created only when you:

- Indicate that you use the IMSADF II Data Dictionary extension on the installation panel
- Choose DL/I database support

The functions done by this member are described in the *IMS Application Development Facility II Version 2 Release 2 Data Dictionary Extension User's Guide*.

DDESAMP (Use Data Dictionary Sample)

This member is created only when you:

- Indicate that you use the IMSADF II Data Dictionary extension on the installation panel
- Choose DL/I database support

The functions done by this member are described in *IMS Application Development Facility II Version 2 Release 2 Data Dictionary Extension User's Guide*.

JCLNLINK (Create LMOD Entries)

This job executes the supplied SMP installation procedure using ADFLINK as SMPJCLIN. This creates LMOD entries in the CDS or target CSI and results in automatic relinking of all executable modules when one module contained in the CDS or target CSI is changed. Link-edit structures are stored in these entries.

Note: All ddnames of included modules must be present in the SMP procedure. This includes non-IMSADF II libraries such as:

- LOAD, in which the IMS language interface module (DFSLI000) resides if DL/I support is selected.
- DDLOAD, in which the IBM Data Dictionary language interface module (DBDWLNKA) resides (if you use it)
- ALOAD, in which the IBM DB2 TSO language interface module (DSNELI or DSNCLI) resides (if you use it)
- RULLIB, a nonSMP-managed library

JCLNLINK results in a condition code of 4.

LOCKWORD (Add Lockword Exit)

This job is executed if you want to add a Sign-On and/or Sign-Off exit to IMSADF II. This topic is discussed in “Installing Sign-On and Sign-Off Exits” on page 77. This job is necessary only if you want to support multiple national languages (see Chapter 9, “National Language Support” for more information).

This job replaces the supplied exits MFC1E01 and/or MFC1E09 with ones you have compiled and linked using the SMP usermod facility.

MFSJOB (Run MFS Utility for IMS IMSADF II-Supplied Screens)

This job places IMSADF II MFS control blocks into the IMSVS.FORMAT library. The job must be complete before IMSADF II can be executed online with IMS.

This job does the screen processing for all IMSADF II screens. Members in IMSADF.SCREENS are input to the procedures.

Notes:

1. Expect a condition code of 4 from the second step of the MFS utility procedure when no index maintenance is done on the IMSVS.FORMAT library.
2. Another condition code of 4 occurs when a DFLD referenced in the MID/MOD is not defined in the FMT. IMSADF II MODs are designed to handle the largest 3270 screens currently available. The FMT describes the actual screen size. Therefore, any terminal type with fewer lines than the maximum now supported shows MFLDs referencing DFLDs that are not present in the FMT. This condition is not cause for alarm. See the IMS library for additional information.

SCREENS (Customize IMSADF II Screens for IMS)

This job invokes installation macro ADFMFS, which produces MFS source statements in the IMSADF.SCREENS library for the base screens supplied by IMSADF II. You can support as many types of 3270 terminals as necessary by correctly completing this macro.

"Changing Terminal Types in IMSADF II" on page 76 gives you complete instructions on how to correctly modify this macro.

See "ADFMFS Macro (Defining Screen Characteristics)" on page 32 for instructions on the coding of this macro.

See Chapter 9, "National Language Support" to find out how to run this job when you install a primary or alternate language.

UCLDEL (Delete ????SYSP Entries Using UCLIN DELETE)

This job deletes from the SMP data sets all entries associated with job ADFOPT. Invoke this job only when the APPLY command finishes in error, as described in "ADFOPT (Customize IMSADF II Installation Options)" on page 52.

Through UCLIN processing, the SYSMOD that added the ????SYSP control block is deleted from the PTS and target zone. The following are also deleted from the target CDS/CSI:

- MOD (????SYSP)
- SRC (????SYSP)
- LMOD (????)

UCLDEL is only run when the ????SYSP control block has not yet been ACCEPTed and, therefore, cannot be RESTORED (since none of the elements has been added to the DLIBS).

Chapter 6. Customizing Existing IMSADF II Systems

This chapter describes the tasks you must do if you decide to recustomize your IMSADF II system after installation. The chapter describes:

- Specifications that can't be changed
- ISPF Installation Dialog CHANGE, DELETE, and CUSTOMIZE options
- Synchronizing ISPF tables for IADF
- Installing changed DEFADF options (you need to execute the ADFUSER job whenever you change DEFADF macro options)
- Running the ADFUSER job on other libraries
- Guidelines for various recustomization situations

Specifications that Cannot Be Changed

A list of specifications that cannot be changed after initial installation, and the reasons why, follows:

ADFID Because the source member name is customized to your initial installation ADFID value

ENVIR Because the only supported environment is IMS.

DB2DB Because this parameter establishes the basic environment in which IMSADF II operates.

TRXTRLR Because this parameter is closely tied to a multiple ADFID environment

MFSTRLR Because this parameter is closely tied to a multiple ADFID environment

ISPF Installation Dialog Recustomization Options

The following subsections describe the CHANGE, DELETE, and CUSTOMIZE options of the ISPF Installation Dialogs.

For information on how to invoke the dialogs and an introduction to the Installation Menu, see “Invoking the ISPF Installation Dialog” on page 34.

CHANGE Option

If you select option 2 (as shown in Figure 22), you invoke the CHANGE process. Before making changes, read Chapter 6, “Customizing Existing IMSADF II Systems,” which describes some common changes you can make and how to do them.

At this time, the same screens that were used for the INSTALL process are presented, letting you change the required values (except protected values). The CHANGE process assumes that the ADFID was previously installed; it issues an error message if that is not the case.

The screens used for this option are scrollable, meaning that the UP and DOWN keys are active and can be used to go back and forth in this sequence of screens.

In viewing the figures in this section, you will see changed values highlighted. The values chosen are by way of example only. They are not necessarily recommended values.

```
ADFINS0 ---------------- IMSADF II ONLINE INSTALLATION --------------
OPTION ===> 2

   ADFID ===> MFC1                                  USERID   - TSOID
                                                    TIME     - 14:08
                                                    DATE     - 97/05/04
   1  INSTALL    - Install IMSADF II
   2  CHANGE     - Change Installation Options
   3  DELETE     - Delete a Version of IMSADF II
   4  CUSTOMIZE  - Customize IMSADF II libraries
   5  TUTORIAL   - Information about IMSADF II Installation
   X  EXIT       - Terminate the Installation


PRESS END KEY TO TERMINATE ONLINE INSTALLATION
```

Figure 22. CHANGE: Installation Options Process

The General Options screen, as shown in Figure 23, is displayed.

```
ADFINS1 --------------- GENERAL OPTIONS ----------------  PAGE 1 OF 5
COMMAND ===>

 ADFID                 ==> MFC1

 Environment           ==> I       (I=IMS)
 ADF Databases in DB2  ==> N       (Y=Use DB2, N=Use DL/I)
 Transaction Trailer   ==>
 V50 Resident ?        ==> Y       (Y/N)
 Format of Dates       ==> U       (U,S,B,E,O)
 Assembler Name        ==> F       (F=Assembler F, H=Assembler H)
 Key Audit Names       ==> S       (S=standard,A=alternate)
 User Exit/SPR Format:     L       (L=load format)
 IADF Feature ?        ==> Y       (Y/N)
 DB2 Extension ?       ==> N       (Y/N)
 Data Dictionary ?     ==> y       (Y/N)
 Programming Language  ==> C       (C=COBOL,P=PLI,A=ASSEMBLER)
 RACF Support          ==> T       (U=RACFUSER,T=TERMINAL,R=RULEGEN)
 End of text Delimiters ==> $$     (Used to signal end of text data)
 Primary Language      ==> E       (E,F,G,J,K,P,S,W or 0-9)
 Alternate Languages ? ==> n       (Y/N)

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 23. CHANGE: General Options

Although you cannot see this in the example, the following fields are protected and cannot be modified, based on your initial INSTALL option choices:

- **ADFID**
- **Environment**
- **ADF Databases in DB2**

The remaining values can be changed, and all appropriate validations are repeated.

Because N is entered beside the **Alternate Languages** entry, the next panel shown is Figure 24. (Otherwise Figure 13 would have appeared for you to update.)

```
ADFINS2 ------ INSTALLATION SCREEN CHARACTERISTICS ------ PAGE 2 OF 5
COMMAND ===>

 ADF SYSTEM ID: MFC1


 Device Name(s)            ==> A2,A4
 Device Type(s)            ==> 2,4
 Device Characteristics    ==> 0,0
 No. of Default Terminals  ==> 2         (1-5)
 MFS Trailer               ==> 1
 Default Printer LTERM     ==> PRINTER
 MFS PRINT statement       ==> N         (G=gen,N=nogen)
 DEV statement PAGE option ==> D         (F=float,D=defn,S=space)
 Format for Literals       ==> D         (L=left,R=right,D=dash)
 MFSUTL Procedure Name     ==> MFSUTL

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 24. CHANGE: Screen Options

Screen characteristics are presented again, and the validations done during installation are repeated.

You are given an opportunity, as shown in Figure 25, to change existing data set characteristics.

```
ADFINS3 ---- INSTALLATION DATA SET CHARACTERISTICS -------PAGE 3 OF 5
COMMAND ===>

 ADF SYSTEM ID: MFC1

 Node for Shared Files   ==> CC.IMSADF          (ADFNODE)
 For Non-shared Files    ==> NEW.IMSADF         (NEWADF)
 Old IMSADF II Node      ==> OLD.IMSADF         (OLDADF)
 IMS/VS Node             ==> IMS130             (IMSNODE)
 IMS/VS Node (PSB/DBD)   ==> TEST               (IMSTEST)
 CICS Node               ==>                    (CICNODE - Ignore)
 ISPF Node               ==>                    (For ISPLINK)
 Data Dictionary Node    ==> DBD50              (For DDLOAD)
 DB2 Node                ==>                    (For DB2 Libraries)
 SMP Node                ==> CC.IMSADF          (SMPNODE)
 PROCLIB Data Set Name   ==> SYS1.PROCLIB       (For IMSADF Procs)
 Rules Library DDNAME    ==> RULES
 No. of BLDL Entries     ==> 128
 Use Composite Modules ? ==> A                  (Y=yes,N=no,A=alt)
 Composite DDNAME        ==> COMP
 No. of BLDL Entries     ==> 64

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 25. CHANGE: Nodes

Next you can change DL/I procedure names or subsystem information, as shown in Figure 26.

```
ADFINSA ----- STORAGE INFORMATION AND MISCELLANEOUS  --- PAGE 4 OF 5
COMMAND ===>

 ADF SYSTEM ID: MFC1

 Available Storage        ==> 0       (0-9999K for Rules Management)
 CICS SPA Storage Type    ==>         (A=AUX, M=MAIN - Ignore)
 Monitor ADF Transactions ==>         (In CICS, Value Logged - Ignore)

 DB2 Information                      If DB2 used for ADF Data Bases
 Subsystem Name           ==>
 Data Base Name for ADF   ==>
 Storage Group            ==>
 High Level Table Node    ==>         (SYSADF)

 Procedure Information
 DBDGEN Procedure Name   ==> TDBDGEN
 PSBGEN Procedure Name   ==> TPSBGEN
 ACBGEN Procedure Name   ==> TACBGEN
 Esoteric Name (DASD)     ==> SYSDA   (for Temporary Space)

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 26. INSTALL: Storage and Subsystem Information

SMP and Work database/SPA information is shown in Figure 27.

```
ADFINS4 -- INSTALLATION WORK DATA BASE/SMP INFORMATION -- PAGE 5 OF 5
COMMAND ===>

 ADF SYSTEM ID: MFC1


 SPA or Work Area Size    ==> 30000      (From 6000 to 64800) WRKAREA
 Use Work Data Base ?     ==> Y          (Y/N)
 Maximum no. of Segments  ==> 5          (From 1-15)          WRKSEGS
 Segment Size             ==> 6000       (At least 2100)      WRKSIZE
 Work Area Reduction ?    ==> N          (Y/N)
 Communication Area Size  ==> 100        (Used at Sign-on Time)

SMP INFORMATION
 Version                  ==> 4          (4 or E)

 SMP/E only:
 GLOBALZONE Data Set Name ==>
 DLIBZONE name            ==>
 TARGETZONE name          ==>

CANCEL to EXIT without saving, ENTER or END to Continue
```

Figure 27. CHANGE: SPA and SMP Options

After the changed information is collected, you are shown Figure 28.

```
ADFINS7 ------------- CUSTOMIZE IMSADF II SOURCE LIBRARIES -----------
COMMAND ===>

                                                    USERID   - TSOID
                                                    TIME     - 14:10

Produce ADFUSER member


       USERMOD name ===>   ADF0002

       SUPMOD  name ===>   ADF0001

JOB STATEMENT INFORMATION:      (No JOBS will be submitted)
  ===> //TSOID    JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*
  ===> //*

Press Enter to Produce ADFUSER member  or END, CANCEL to exit
```

Figure 28. CHANGE: Produce ADFUSER Member

Enter the SMP usermod name that you are superseding (SUPMOD) and the name of your new usermod (USERMOD).

When you press ENTER, IMSADF.JCLLIB(ADFUSER) is customized and ready for you to submit. If you exit before customization, your changed values are **not** saved.

After job ADFUSER is produced, the CHANGE option is complete. The changes you have made may require that you recustomize one or more libraries. "CUSTOMIZE Option" on page 67 discusses the next option you should select on the menu.

The CHANGE option also updates row IMSADF.ADFTLIB(PRODUCT) in the ISPF table. This row contains the options you just selected. See "Synchronizing ISPF Tables for IADF" on page 71 for more information about installation-related tables. Notify your IADF administrator that you have changed installation options, so that application developers can reset their IADF profiles if necessary.

DELETE Option

Select this option (with extreme care) to delete an installed version (ADFID) of IMSADF II. The DELETE option deletes rows from ISPF tables containing options and administrative users registered for the ADFID. It does not delete any libraries or databases for this ADFID. See "Synchronizing ISPF Tables for IADF" on page 71 for more information about installation-related tables.

There are two situations in which you need to use DELETE:

- You made an error on the dialog panels and will reinstall the ADFID.
- The ADFID was installed normally, IADF was used to develop applications, and now you want to delete everything associated with this ADFID.

In this manual, the first situation is assumed, and no SYSIDs are deleted from the SYSTEMS table as a result of DELETE.

Note: If you want to delete the ADFID **without** reinstalling (and you have established SYSIDs using IADF), do the following using the *IMS Application Development Facility II Version 2 Release 2 Interactive ADF Administration Guide*, as appropriate:

1. Log on to TSO and invoke IADF.
2. Delete all SYSIDs for the ADFID.
3. Exit from IADF.
4. Invoke the ISPF Installation Dialogs DELETE option to delete this ADFID.

If you don't follow this process, there will be inaccessible rows in the SYSTEMS table.

Select the DELETE option as shown in Figure 29.

```
ADFINS0 ---------------- IMSADF II ONLINE INSTALLATION --------------
OPTION ===> 3

   ADFID ===> MFC1                                  USERID   - TSOID
                                                    TIME     - 14:09
                                                    DATE     - 97/05/04
   1  INSTALL    - Install IMSADF II
   2  CHANGE     - Change Installation Options
   3  DELETE     - Delete a Version of IMSADF II
   4  CUSTOMIZE  - Customize IMSADF II libraries
   5  TUTORIAL   - Information about IMSADF II Installation
   X  EXIT       - Terminate the Installation


PRESS END KEY TO TERMINATE ONLINE INSTALLATION
```

Figure 29. DELETE: Installed ADFID

Figure 30 appears, listing all the ADFIDs you have installed.

```
ADFINS8 ---------------------- DELETE ONLINE VERSIONS ---------------
COMMAND ===>                                      SCROLL ====>  PAGE


Line Command: D delete
CMD      ADFID        Created by:        Date     Time
-        ----         -----------        --------------
'        MFC1         TSOID1             97 09 06 21:45
d        MFC2         TSOID2             97 07 25 15:01
'        ADF2         TSOID2             97 07 25 15:02
'        MFC6         TSOID1             97 05 03 13:22
'        MFC7         TSOID1             97 05 03 13:24
****************************** BOTTOM OF DATA **********************
```

Figure 30. DELETE: List of Installed ADFIDs

The list has the ADFID, the TSO user who installed it, and a date/timestamp.

Select a value by entering `d` under the CMD column beside the appropriate row.

Figure 31 lets you confirm the deletion.

```
---------------C O N F I R M  D E L E T I O N --------------------
COMMAND ===>

 An Online version of IMSADF II is going to be deleted:

 MFC2

 Date Created:  07/25/97

 Time Created:  15:01:21

 By:            TSOID2



 Press ENTER key to CONFIRM DELETE.

 Press END key to CANCEL DELETE.
```

Figure 31. DELETE: Deletion Screen

Press ENTER to confirm the deletion.

The next screen you see is the IMSADF II main screen with the deletion message on the third line.

CUSTOMIZE Option

This option should be used:

- After a change is made to one or more values in the DEFADF macro using option 2 (see “CHANGE Option” on page 60), or
- As a result of applying maintenance to IMSADF II (see Chapter 10, “Post-SMP Maintenance”).

This process lets you recustomize:

- Selected members of one library,
- A specific member, or
- A complete library

The customization process can be lengthy if many members or IMSADF.JCLLIB are tailored. The CUSTOMIZE option is chosen, as shown in Figure 32.

```
ADFINS0 ---------------- IMSADF II ONLINE INSTALLATION --------------
OPTION ===> 4

   ADFID ===> MFC1                                    USERID   - TSOID
                                                      TIME     - 14:09
                                                      DATE     - 97/05/04
   1  INSTALL    - Install IMSADF II
   2  CHANGE     - Change Installation Options
   3  DELETE     - Delete a Version of IMSADF II
   4  CUSTOMIZE  - Customize IMSADF II libraries
   5  TUTORIAL   - Information about IMSADF II Installation
   X  EXIT       - Terminate the Installation


PRESS END KEY TO TERMINATE ONLINE INSTALLATION
```

Figure 32. CUSTOMIZE: Selected Libraries and Members

Then Figure 33 is presented.

```
ADFINS6 ------------- CUSTOMIZE IMSADF II SOURCE LIBRARIES ----------
OPTION ===> 1

                                                    USERID   - TSOID
                                                    TIME     - 14:11
   1  LIBRARY    - Customize a complete source library
   2  MEMBER     - Customize selected members

   Library type ===> d              (J=JCL,D=Dbdsrc,P=Psbsrc,R=Rules)
   Member name  ===>                (Blank for member selection list)


JOB STATEMENT INFORMATION:      (Verify before proceeding)
  ===> //TSOID    JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*
  ===> //*

Enter Cancel or Press END key to exit
```

Figure 33. CUSTOMIZE: One Library

On this screen, enter a one-character abbreviation for the target library. The target libraries known to the system are JCL, DBDSRC, PSBSRC, and RULES.

In this example, a complete library (DBDSRC) is selected and the JOB and routing information, when appropriate, can be added to the output produced.

Once the process is complete, Figure 34 is presented.

```
--- CUSTOMIZE IMSADF II SOURCE LIBRARIES ----- 4 members customized
OPTION ===>

                                                    USERID   - TSOID
                                                    TIME     - 14:11
   1  LIBRARY    - Customize a complete source library
   2  MEMBER     - Customize selected members

   Library type ===> D             (J=JCL,D=Dbdsrc,P=Psbsrc,R=Rules)
   Member name  ===>               (Blank for member selection list)


JOB STATEMENT INFORMATION:      (Verify before proceeding)
  ===> //TSOID    JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*
  ===> //*

Enter Cancel or Press END key to exit
```

Figure 34. CUSTOMIZE: Library (Completed)

Figure 34 shows the total number of customized members (four in this case).

If selected members are to be customized, rather than an entire library, choose option 2, as shown in Figure 35.

```
ADFINS6 ------------- CUSTOMIZE IMSADF II SOURCE LIBRARIES ----------
OPTION ===> 2

                                                    USERID   - TSOID
                                                    TIME     - 14:11
   1  LIBRARY    - Customize a complete source library
   2  MEMBER     - Customize selected members

   Library type ===> j             (J=JCL,D=Dbdsrc,P=Psbsrc,R=Rules)
   Member name  ===>               (Blank for member selection list)


JOB STATEMENT INFORMATION:      (Verify before proceeding)
  ===> //TSOID    JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*
  ===> //*


Enter Cancel or Press END key to exit
```

Figure 35. CUSTOMIZE: Selected Members

When you choose option 2, you do not have to enter the member name, although you do have to name the library type.

If you leave the member name blank, Figure 36 is presented.

```
ADFINS5 -- MEMBERS CONTAINED IN JCL LIBRARY ------------- ROW 5 OF 62
COMMAND ===>                                          SCROLL ===>  PAGE
Line Command: S select
Select NAME               Description
       ADFACBS    ACBGEN for Supplied Transactions
       ADFDATAS   Load IMSADF II Data Bases w/Supplied Data
       ADFDBDS    DBDGEN for IMSADF II Data Bases
 s     ADFLINKS   Links IMSADF II Execution Modules
       ADFOPTS    DEFADF Macro / USERMOD
       ADFPSBS    PSBGEN for IMSADF II Supplied Transactions
 s     ADFRELDS   RELOAD SMP Data Set Entries
       ADFSTG1S   Input for IMS/VS Stage 1 SYSGEN
 s     ADFULNGS   USERMOD for User Language(NLS Support)
 s     ADFUNLDS   UNLOAD SMP Data Sets
       ADFUSERS   SMP USERMOD for Changing Options
       ADFWORKI   Initialize Work Data Base
       COPYPRCS   Copies IMSADF II procedures to SYS1.PROCLIB
       DBUPDATS   Update existing D/B with data for current release
       DDEECI     Data Dictionary Extensibility Commands for Model
       DDEECI5    Data Dictionary Commands (Rel. 5)
       DDESAMPC   Data Dictionary Sample
```

Figure 36. CUSTOMIZE: Selection Screen for a Specific Library

Figure 36 is an alphabetical list of the members contained in the specific library.

Here you can choose one or more members by entering S (selection) in the left-most column, as shown. After the members are customized, they are shown to you in browse mode sequentially, unless you pressed END instead of ENTER after your final S (selection) above. In any case, you are again shown Figure 37 with an informational message giving the total number of customized members.

```
--- CUSTOMIZE IMSADF II SOURCE LIBRARIES ----- 4 members customized
OPTION ===>

                                                    USERID   - TSOID
                                                    TIME     - 14:13
   1  LIBRARY    - Customize a complete source library
   2  MEMBER     - Customize selected members

   Library type ===> J         (J=JCL,D=Dbdsrc,P=Psbsrc,R=Rules)
   Member name  ===>           (Blank for member selection list)


JOB STATEMENT INFORMATION:      (Verify before proceeding)
  ===> //TSOID    JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*
  ===> //*

Enter Cancel or Press END key to exit
```

Figure 37. CUSTOMIZE: Selected Members Completed

If you want to tailor more members, continue to select them. When you press the END key, the CUSTOMIZE process is complete.

If you want to customize a known member (see Figure 38), enter the member name and option 2 for customization.

```
ADFINS6 --------------- CUSTOMIZE IMSADF II SOURCE LIBRARIES -------
OPTION ===> 2
                                                      USERID   - TSOID
                                                      TIME     - 09:47
   1  LIBRARY    - Customize a complete source library
   2  MEMBER     - Customize selected members

   Library type ===> J              (J=JCL,D=DBDSRC,P=PSBSRC,R=Rules)
   Member name  ===> ADFDATAS       (Blank for member selection list)


JOB STATEMENT INFORMATION:      (Verify before proceeding)
  ===> //TSOID    JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*
  ===> //*

Enter Cancel or Press END key to exit
```

Figure 38. CUSTOMIZE: Specific Member

When the member is customized, it is shown to you in browse mode, followed by the original Figure 38 with the informational customization message.

Synchronizing ISPF Tables for IADF

There is a relationship between ISPF Installation Dialogs and IADF that affects application development. Linkage is through three ISPF tables produced or updated through the dialogs. The tables, which exist in IMSADF.ADFTLIB, are as follows:

PRODUCT This table has one row per ADFID and contains the options you chose on the screens for the INSTALL or CHANGE options. The table's key is ADFID. The first time you execute the ISPF Installation Dialogs, this table is created for you.

SPECUSER The key of this table contains the ADFID; the table contains a row of all IADF Administrative Users for the ADFID. The first time you execute the ISPF Installation Dialogs, this table is created for you. Each time you INSTALL an ADFID, your TSO userid is added as an Administrative User Level 3. For more information about Administrative Users, see *IMS Application Development Facility II Version 2 Release 2 Interactive ADF Administration Guide*.

SYSTEMS This table contains critical information about the SYSIDs you installed under IADF for all ADFIDs. The first time you execute the ISPF Installation Dialogs, this table is created for you. The key of this table contains both the ADFID and SYSID. Each row has information about data sets that are designated for the SYSID. This

table can expand to many rows. If SYSTEMS is destroyed, IADF has no knowledge of any SYSID under its development.

The INSTALL and CHANGE options update the PRODUCT table. The DELETE option removes the affected row from the PRODUCT and SPECUSER tables.

PRODUCT establishes defaults used as input to the Rule Generator, and provides high-level library node names and cataloged procedure names. Developers can modify these values when they establish an IADF profile, stored in their ISPPROF data set.

It is important that the PRODUCT table be synchronized with jobs ADFOPT(INSTALL) or ADFUSER(CHANGE), since application developers using IADF rely on the accuracy of this table. When you do either of these ISPF Installation Dialogs options, you must run the affected job successfully to ensure consistency. You should then notify your IADF administrator so that developers can reset their profiles as appropriate.

Installing Changed DEFADF Options (ADFUSER Job)

You need to run this job whenever you change DEFADF options.

The SMP USERMOD facility makes changes to ????SYSP by invoking the DEFADF macro. Change the keyword values when necessary to replace the values originally installed.

Figure 39 shows the format of the usermod.

```
//SMPPTFIN  DD  *
++USERMOD (ADF0002).
++VER(P115)  FMID(HGE2202)  SUP (ADF0001).
++SRC(????SYSP) DISTLIB(ADFMAC) DISTMOD(ADFMOD).
     DEFADF macro invocation

            change the keyword values to yours

     END
/*
//SMPCNTL   DD  *
```

Figure 39. Usermod for Changing DEFADF Macro Options

Since this is an SRC replacement, sequence numbering is not important. There is a SUP operand on the VER statement for the usermod. The SUP operand refers to the usermod name you used at initial installation in ADFOPT.

After you RECEIVE and APPLY this usermod, SMP assembles ????SYSP and automatically relinks every load module that names that member. This makes all your IMSADF II execution time modules current.

If you leave this usermod in APPLY status, you can reuse its number each time you change options; to do this:

- RESTORE and REJECT the usermod
- RECEIVE and APPLY the usermod

You can ACCEPT this usermod, but you will have to enter new USERMOD/SUPMOD values the next time you want to change options.

Changes Requiring Additional Steps

When you change an existing option, you may have to recustomize some or all of the following libraries using the CUSTOMIZE option on the Installation Menu:

IMSADF.JCLLIB
IMSADF.RULES.SOURCE
IMSADF.DBDSRC
IMSADF.PSBSRC

Figure 40 lists each option or DEFADF macro keyword, and identifies the libraries that should be customized when an option is changed.

In some cases, a keyword may affect one member of a library or a limited set of members. The final table column identifies affected members (or provides general notes as appropriate). The name of the input member generally appears in the last table column, with an *S* appended (for example, ADFLINKS).

Figure 40 (Page 1 of 2). DEFADF and Online Options

KEYWORD	JCL	RULE	DBD	PSB	MEMBER or NOTES
ACBGEN	x				ADFACBS
ADFID	x	x	x	x	Cannot change
ADFDB2	x				
ADFNODE	x				
CICNODE	Ignore				
DATA DICT	x				ADFLINKS, DDE* members, ADFUNLODS
DB2	x				ADFLINKS, PROCGS, ADFUNLODS
DB2DB	x	x	x	x	Cannot change
DB2ID	x				ADFTB* members
DB2NODE	x				ADFTB*, ADFLINKS
DBDGEN	x				ADFDBDS
DDNODE	x				ADFLINKS
DEVCHRS		x			
DEVNAME	x	x			Screens and rules
DEVTYPE	x	x			Screens and rules
DLIBZONE	x				SMP jobs
ENVIR	x	x	x	x	Cannot change
GLOBALZONE	x				SMP jobs
IADF	x				ADFISPU
IMSNODE	x				
IMSTEST	x				

Figure 40 (Page 2 of 2). DEFADF and Online Options

KEYWORD	JCL	RULE	DBD	PSB	MEMBER or NOTES
ISPNODE	x				ADFISPU
MCTID	Ignore				ADFCMCTS
MFSTRLR	x				Do not change
MFSUTL	x				MFSJOBS
NEWADF	x				
PROCLIB	x				COPYPRCS
PSBGEN	x				ADFPSBS
SMPNODE	x				PROCGS, SCREENS, Ignore
SMP	x				SMP jobs
STOGRP	x				ADFTB* members
SYSADF	x	x			
TARGETZONE	x				SMP jobs
TRXTRLR	x		x	x	Do not change
USRLANG	x				
WRKAREA	x		x	x	
WRKSEGS	x		x	x	
WRKSIZE	x		x	x	

There are other DEFADF parameters required on the ISPF panels; however, only those keywords listed in Figure 40 on page 73 affect library customization.

You should recustomize job SCREENS if any of the following values are changed in ADFMFS:

USRLANG
LISTING
PAGE (DEVNAME and DEVTYPE are listed previously)

Changes Requiring No Additional Steps

There are certain installation options you can change that require no additional steps after you execute job ADFUSER; these options (listed below) require no customization of IMSADF II target libraries:

ALTLANG Although this parameter requires no customization of libraries, it does require additional steps as described in Chapter 9, “National Language Support.”

ASMBLR

COMMLEN Because users map this area in their application programs, a change in this value could require application changes.

COMPMBL Make sure enough main storage is available in the message region. Add the product of 66 times the number of entries.

COMPMDD

COMPMOD If you change COMPMOD to YES, be sure you have Primary and Secondary Option Menu Rules IMSADF II preloaded by IMSADF II. It is a good idea to test your composites before implementing this change in production, since transactions could terminate if rules are missing in the composite rules load modules.

DATEFMT

DELIM Be sure this delimiter does **not** normally appear in your batch input.

KANAME Be sure all relevant audit keys are changed if necessary, as this is the name built at execution time (unless the Rule Generator overrides it).

NCPRINT

OLDADF

RACF Before you change your option to RACFUSER (see “DEFADF Macro (Defining Basic IMSADF II Characteristics)” on page 15), add valid RACF userids for supplied transactions, particularly Data Base Administration (SYSID=????), to your Sign-On Profile database. Otherwise, once the option is changed you will not be able to access those transactions (or SAMP), unless *999999* is a valid RACF userid at your installation.

RULESBL Make sure enough main storage is available in the message region. Add the product of 66 times the number of entries.

RULESDD

RULESVS Make sure enough main storage is available in the message region.

SPATS

TRMDFLT You can change the NUMBER of default sets of terminal characteristics saved for you by IMSADF II.

Also, if those sets are **currently** installed but are not currently defaults, you can specify additionally the DEVNAME, DEVTYPE, and DEVCHRS associated with your desired default. However, if the above values are not generated for your IMSADF II system, you must add them based on instructions in “Changing Terminal Types in IMSADF II” on page 76.

V50RES

WRKREDU Make sure enough main storage is available in the message region. Add the value of WRKAREA to your storage requirements.

Guidelines for Various Recustomization Situations

The following sections describe various, specific changes you may want to make.

Changing the IMSADF II SPA/Work Database

This section tells you how to change the SPA or Work database segment size or number in different environments.

IMS DL/I Environment

Besides describing how to change segment size and number, the following procedure also describes how to change from the SPA to the Work database.

1. Change your WRKSIZE, WRKSEGS, and WRKAREA values. Run ADFUSER to completion.
2. Customize all libraries except IMSADF.RULES.SOURCE. This ensures that the size and number of your work segments are correct throughout the macros and JCL that use this information.
3. Run job ADFDBD. However, only select the DBD for the Work database. As a result of the previous step, all symbolic references should now be current.
4. Rerun all IMSADF II PSBs, as found in job ADFPSB. Remember, references to the number of segments are now customized.
5. Run job ADFACB.
6. Run job ADFWORKI to reinitialize the Work database.
7. Redo the conversational application PSBs to reflect the changes you have made. This task is made as automatic as is possible when you use the IMSADF II macro BUILDT07. After recustomization of PSBSRC, all symbolic references became the values you specified, so recreate those PSBs and ACBs using your own procedures.
8. If you are changing from the IMS SPA to the Work database, you must do an IMS system generation. Add the DATABASE macro for the IMSADF II Work database, and change the SPA size in the TRANSACT macro to 28.

All Other Environments

The only value you can change is WRKAREA.

Invoke the ISPF Installation Dialogs CHANGE option, changing your WRKAREA value; then run ADFUSER to completion.

Changing Terminal Types in IMSADF II

To add a new terminal type after you have installed IMSADF II, you must change:

- The system-supplied base screens
- The sample rules
- Any IMSADF II applications to be executed on the new terminal type

This section tells you how to change the base screens and the sample rules. The *IMS Application Development Facility II Version 2 Release 2 Application Development Guide* tells you how to modify IMSADF II applications for use on other types of terminals.

1. Change the DEVNAME, DEVTYPE, and DEVCHRS keywords. Then run ADFUSER to completion. This changes the terminal type for the distributed system rules so that those transactions execute properly on the new terminals.
2. CUSTOMIZE the following:

- SCREENS members of IMSADF.JCLLIB. Based on your subsystem choice, SCREENS invokes ADFMFS for the terminal types you specified. See “ADFMFS Macro (Defining Screen Characteristics)” on page 32 for complete details on these parameters.
- IMSADF.RULES.SOURCE library, which makes the change to the system rules.

3. Execute SCREENS to create your base screen source.
4. Run the Rule Generator to create the screen source for IMSADF II-supplied transactions using job ADFLINK. Select only the parts of ADFLINK that invoke the Rule Generator (????G procedure). When this is done, screen source from all system transactions and the sample problem apply to all of your terminal types.
5. Run job MFSJOB to create your actual formats.
6. For each of your transactions for processing on the new terminal type, do the following:
   a. Run the Rule Generator utility to create new Sign-On, Primary Key Selection, and Data Display source after you add the new terminal type.
   b. Invoke the MFS utility to add the new formats to the subsystem screen library.

Installing Sign-On and Sign-Off Exits

This section tells you how to install your own sign-on (Lockword) and sign-off exits.

IMSADF II supplies a sign-on exit (MFC1E01) and a sign-off exit (MFC1E99). These supplied exits are dummy exits. You are expected to replace them with your own exits after you code them. There should be no IBM maintenance to either of the modules you are replacing. See the *IMS Application Development Facility II Version 2 Release 2 Application Development Guide* and the *IMS Application Development Facility II Version 2 Release 2 Application Development Reference* for information on how to code these exits.

Use one of the two following procedures, based on the language in which you wrote the exits, to install the exits.

COBOL or Assembler Exits

Compile your exits and link them to IMSADF.RULLIB, including any language sub-routines your exits use.

To incorporate your exits into the sign-on (????TOM) and sign-off (????T99) modules, use job LOCKWORD. This job places the usermod shown in Figure 41 on your system.

```
//SMPPTFIN DD *
++USERMOD (ADF0100).
++VER(P115) FMID(HGE2102).
++MOD(MFC1E01) LKLIB(RULLIB).
++MOD(MFC1E99) LKLIB(RULLIB).
/*
```

(COBOL or Assembler)

Figure 41. Usermod for Installing User Sign-On and Sign-Off Exits

Notes:

1. You do not have to code both exits if you do not need them.
2. You can choose another name for your usermod.
3. MFC1E01 and MFC1E99 are specifically named modules; they are **not** customized by ADFID.

RECEIVE and APPLY this usermod. The APPLY relinks the ????TOM, ????T99, and ????BXX load modules. Since there is no service to these modules, you can leave them in APPLY for ease in making your own changes to the exits.

Since the exits are now located in ????BXX, you do not have to relink ???BDXX (the batch transaction driver) when you invoke them. See the *IMS Application Development Facility II Version 2 Release 2 Application Development Reference* for additional requirements when you use sign-on and sign-off exits with batch transaction driver rules.

PL/I Exits

When you use PL/I exits, you must add an interface module for these exits. Figure 42 and Figure 43 show the linkage editor control statements you must use to implement non-main procedure PL/I exits. In each figure, replace the ???? with your ADFID; you can supply the name of your library and your exit name.

Note: You are encouraged to use the supplied exit names MFC1E01 and MFC1E99. If you use them, replace the first two statements below with the third one.

```
INCLUDE  yourlib(yourprog)
CHANGE   MFC1E01(yourprog)
     or INCLUDE yourlib(MFC1E01)
              replaces the above two statements
INCLUDE  ADFMOD(MFC1E01P).
CHANGE   MFC1E01C(MFC1E01P)
INCLUDE  SYSLMOD(????TOM).
MODE     AMODE(31)
ENTRY    MFC1TOM
NAME     ????TOM(R)
/*
```

Figure 42. Linkage Editor Statements for Sign-on Exit (PL/I)

```
INCLUDE  yourlib(yourprog)
CHANGE   MFC1E99(yourprog)
     or INCLUDE yourlib(MFC1E99)
              replaces the above two statements
INCLUDE  ADFMOD(MFC1E99P).
CHANGE   MFC1E99C(MFC1E99P)
INCLUDE  SYSLMOD(????T99).
MODE     AMODE(31)
ENTRY    MFC1T99
NAME     ????T99(R)
/*
```

Figure 43. Linkage Editor Statements for Sign-off Exit (PL/I)

Chapter 7. Multiple IMSADF II Systems

This chapter describes the multiple ADFID environment. It describes this environment primarily from the standpoint of transaction subsystem environmental constraints. If you decide to install more than one ADFID, see “SMP Considerations for the Multiple ADFID Environment” on page 85.

Multiple IMSADF II systems can exist under the control of a single IMS control region.

Note: Before you choose an ADFID, please review the following pages carefully.

General Considerations

You may be approaching this topic with one of several goals in mind:

- You want separate copies of one IMSADF II system, so that you can do testing.
- You are installing a new release of IMSADF II, while you have another in production mode.
- You want to apply maintenance to one copy of IMSADF II without putting it on your production system at the same time.
- You have IMSADF II installed on one subsystem (IMS) and want to install it on another.
- You want to replace your current system with a later IMSADF II release using the same ADFID.

Generating multiple IMSADF II systems involves additional effort on your part. How much effort depends on how separate you want your two IMSADF II systems to be. Generally, the greater the separation, the simpler the procedures you must follow.

Throughout this chapter, installation of a new ADFID assumes that you follow the instructions in this guide, the same as you did for the original ADFID. A complete installation is not necessary if you share SMP data sets and common libraries among ADFIDs.

Installing Multiple ADFIDs

This subsection describes multiple ADFIDs only from the standpoint of the ISPF Installation Dialogs.

The ISPF Installation Dialogs can save multiple sets of installation options. You install multiple ADFIDs much the same as was described previously in the INSTALL option, with several exceptions.

When you see the main installation screen, enter your new ADFID and select INSTALL. You will see an information message indicating that there is an ADFID mismatch; in other words, the ADFID you entered does not match the one installed. This gives you the opportunity to exit if you merely miskeyed the ADFID.

Go through the installation screens as you did during the initial install.

When you come to the customization screen, you need to reallocate your four target library ddnames if you have not previously done so:

- PSBSRC - IMSADF.PSBSRC
- DBDSRC - IMSADF.DBDSRC
- JCLLIB - IMSADF.JCLLIB
- RULSRC - IMSADF.RULES.SOURCE

Otherwise, the members in your existing libraries are overlayed.

When customization is complete, continue with your installation of another ADFID by submitting the required jobs.

DEFADF Parameters

During installation job ADFOPT, you can specify the DEFADF keywords in Figure 44. Specifying these values and successfully completing the installation process ensures that the requirements of uniqueness for IMS control blocks, databases, and libraries are handled correctly for IMSADF II-supplied dynamic rules databases and transactions. Next, you must decide how many transactions of your own to run dynamically and create IMS control blocks for them.

The parameters in Figure 44 particularly affect the multiple IMSADF II environment. The recommendations in the figure ensure maximum flexibility in your environment. The recommendations assume that you run both ADFIDs under one IMS control region. They also assume that all your transactions, as well as all the supplied transactions, run under both ADFIDs and that you may be doing development involving screen differences on those transactions.

Figure 44. DEFADF Parameters in the Multiple IMSADF II Environment

ADFID	Make the first two characters different from the original ADFID.
TRXTRLR	Use this parameter so the same transaction can run under both ADFIDs. **If you share SMP data sets** and common IMSADF II libraries with more than one ADFID, this parameter must be different for each ADFID. See “Sharing SMP Data Sets Among Multiple ADFIDs” on page 86 for an explanation.
MFSTRLR	Use this parameter to ensure that formats from one ADFID do not overlay formats from another **if** you may be changing screens of a transaction that runs on both ADFIDs.
RULESDD and COMPMDD	Make these parameters different from those for your first ADFID so that you can control the search order for rules. For example, COMMON00 and KEYAUDIT can be different for each ADFID. You must specify the DD values to ensure correct selection by the proper ADFID. If you want, you can still use the same physical libraries as objects of those ddnames, as you might for your composite rules load modules library. If you link your transaction drivers to IMSADF.ADFLOAD for each ADFID, your drivers are retrieved through STEPLIB.

Another option change requiring reinstallation of an existing ADFID, if not installation of another ADFID, is the database access method (DB2DB).

IADF Requirements

You can use one version of IADF to develop multiple ADFIDs for the same IMSADF II release. The system libraries for ISPF have the prefix ADFNODE and have no customization requirements. Use one set of those libraries, and add the new ADFLOAD library to the ISPLLIB concatenation.

The tables found in IMSADF.ADFTLIB can contain the same information for both ADFIDs in one copy of PRODUCT, SPECUSER, and SYSTEMS. See *IMS Application Development Facility II Version 2 Release 2 Interactive ADF Administration Guide*, "Preparing TSO/ISPF to Use IADF," for specific instructions.

IMS Requirements

IMS requirements center on the following control block libraries (especially DL/I): ACBLIB, PSBLIB, DBDLIB, and FORMAT. System definition requirements are the same requirements you have any time you duplicate DL/I applications to run them in the same IMS control region; the requirements are not unique to IMSADF II. However, IMSADF II offers some aids to help with this effort for the supplied transactions and control blocks; the aids are in the DEFADF macro and the installation process itself.

A brief review: To fulfill IMS library requirements, when you install a new ADFID you have new DBD names and dynamic rules databases. Your load module names have your ADFID as a prefix (with a few exceptions). The installation process provides unique PSBs and ACBs.

Screen Requirements

Installation job SCREENS produces the following screens:

- Primary and Secondary Option Menu
- Secondary Key Selection
- Error

These screens are used across the IMSADF II system. Jobs ADFLINK and MFSJOB create sample problem, database administrator, and IMSADF II-supplied database update rules and screens.

Other concerns center on the IMS libraries FORMAT, ACBLIB, and PSBLIB, and on the ease with which you can run transactions you created under one IMSADF II system on another. DBDLIB is not considered, since those changes occur during installation of IMSADF II, and they will be present when your ACBGENs are done.

Possible Combinations of Multiple IMSADF II Systems

This section describes:

- Using the same ADFID on the same IMS control region when there is a release level change
- Using the same ADFID on multiple IMS control regions (for example, test and production)
- Using different ADFIDs on multiple IMS control regions

- Using different ADFIDs on the same IMS control region (describes two different cases)

Using the Same ADFID for Release Level Changes

This really is not a multiple IMSADF II system, since one release is replacing another **without** an ADFID change. The normal installation process replaces all the supplied control blocks. Your transactions execute without changes. You have new product modules to replace the prior release, but your IMS system definition remains unchanged. Your supplied DBDs, PSBs, and ACBs are replaced at installation, and since your ADFID is the same as before, your application PSBs/ACBs are still correct.

Note: This is the case **only** if the installation options you specified in jobs ADFOPT and SCREENS remain the same as they were in the previous release, especially in the case of Work database parameters. If these options have changed, follow the instructions in “Changing the IMSADF II SPA/Work Database” on page 76.

If you want, you can regenerate some of your current transactions to take advantage of additional functions.

Using the Same ADFID on Multiple IMS Control Regions

This situation commonly exists when there is a test version of the control region separate from the production version. The IMS system definition is the same for transaction and database information. All user transactions and screens run equally well on both systems, without change.

An exception occurs when you select DB2 as your IMSADF II database access method. See “Separate DB2 Tables” on page 85.

Using Different ADFIDs on Different Subsystems

In this case, your application transactions are valid without change, except for your sign-on screens. This is because the Rule Generator creates a literal in the sign-on screen, based on the ADFID; the literal is the name of the scheduled IMS transaction code, for example, the ADFID (the first four characters) followed by T01.

Suppose, for example, that you have two ADFIDs, and the original ADFID is MFC1. If you use the same format on the IMS system with the new ID (for example, ADF2), you will schedule MFC1T01, which is not in your IMS system definition for the new ADFID. You must execute the Rule Generator to generate a new sign-on screen and to link-edit your transaction drivers (which must contain the desired ADFID).

Before installing IMSADF II, copy the FORMAT library for your current IMSADF II transactions to the FORMAT library for your separate control region. Install IMSADF II. Your installation-supplied screens are placed in that library. Then regenerate the sign-on screens for your current transactions.

Your system definition is different for each control region; it carries only the names of one ADFID's database, PSBs, and transaction codes for the supplied transactions. No change is required in the system generation for your existing IMSADF II transactions.

You must recreate all your user PSBs and ACBs. The names of those blocks do not change, but the PCB names of the IMSADF II dynamic rules databases are different. (Using the BUILDTxx macros may be helpful here; if you point to the new IMSADF II data sets, the proper database names are supplied.)

Note: PSBLIB, ACBLIB, and FORMAT libraries **must** be separate or the members created under your first ADFID are overlayed by your second ADFID.

Follow these steps:

1. Copy formats for all current IMSADF II transactions to the corresponding library for the separate control region.
2. Install IMSADF II (installation formats are placed in the library from step 1).
3. Link-edit your transaction drivers with the new Rule Generator utility, specifying your link-edit options and `SIGNON=YES`. The Rule Generator automatically uses the ADFID you specified during installation to create your new sign-on screens. Incorporate your sign-on screens with the MFS utility.
4. Redo your PSBs. By using BUILDTxx macros and pointing at your new ADFID's PSBSRC, you acquire the new IMSADF II dynamic rules database names, without having to do the changes manually.
5. Build your ACBs.
6. Rebind your DB2 application plans, and grant authority in that environment.

Using Different ADFIDs on the Same IMS Control Region

In this instance, you have only one FORMAT, PSBLIB, and ACBLIB. Extra effort is required to ensure the uniqueness that IMS requires of an application program. Two cases are discussed below.

Case One: This case applies when there are two ADFIDs whose first two characters are unique (for example, MFC1 and ADF2).

In installation job ADFOPT, specify your new ADFID. You must also code TRXTRLR, which the Rule Generator uses to create unique IMS transaction codes, PSB names, and sign-on MOD names. As you progress through the installation steps, you satisfy the requirements for uniqueness for the supplied DL/I control blocks and screens.

Note: You may also be required to use MFSTRLR if you want the display screens for the new ADFID to be different from those for the original IMSADF II system (for example, if you have a development system and a production system).

You must do the following to be able to run your existing IMSADF II transactions under the new ADFID:

1. Link-edit your transaction drivers with the new Rule Generator utility, specifying your link-edit options and `SIGNON=YES`. The Rule Generator automatically uses the transaction trailer you specified during installation to create your new sign-on screens and transaction name.
2. Incorporate your sign-on screens with the MFS utility.
3. Redo your PSBs, appending your TRXTRLR value to your PSB name. By using the BUILDTxx macros and pointing at your new ADFID's PSBSRC, you acquire the new IMSADF II system database names without having to do the changes manually.

4. Build your ACBs, appending your TRXTRLR value to your ACBs.
5. Rebind your DB2 application plans, and grant authority in that environment.
6. Update your subsystem system definition. This consists of adding the TRANSACT and APPLCTN information to your IMS stage I system definition and appending the TRXTRLR value to each.

Case Two: This case applies when there are two ADFIDs whose first two characters are the same (for example, MFC1 and MFC2).

In installation job ADFOPT, specify your new ADFID. You must also code TRXTRLR, which the Rule Generator uses to create unique IMS transaction codes, PSB names, and sign-on screens. MFSTRLR is another keyword specified in ADFOPT. This keyword keeps the installation-supplied and current user screens from overlaying the screens for your first ADFID. Screen naming conventions for IMSADF II sometimes use no more than the first two letters of the major system ID; the applications supplied to you (for example, in a release change) may have different screen formats. If MFSTRLR is not specified, the old screens are overlayed with the new. As you progress through the installation steps, you satisfy the requirements for uniqueness for the supplied DL/I control blocks and screens. To be able to run your existing IMSADF II transactions under the new ADFID, do the following:

1. Link-edit your transaction drivers with the new Rule Generator utility, specifying your link-edit options and `SIGNON=YES`. The Rule Generator automatically uses the transaction trailer you specified during installation to create your new sign-on screen and transaction name.
2. In addition to having unique transaction codes and sign-on screens with TRXTRLR as the last character, you must regenerate your Primary Key Selection and Data Display screens with the specified MFSTRLR. This ensures consistency within the major system ID. The Rule Generator automatically updates screens created during a run with MFSTRLR. If all screens were not created with the same MFSTRLR, results could be unpredictable.
3. Incorporate your sign-on screens with the MFS utility.
4. Redo your PSBs, appending your TRXTRLR value to your PSB name. By using the BUILDTxx macros and pointing at your new ADFID's PSBSRC, you acquire the new IMSADF II system database names without having to do the changes manually.
5. Build your ACBs, appending your TRXTRLR value to your ACB.
6. Rebind your DB2 application plans, and grant authority in that environment.
7. Update your subsystem system definition. This consists of adding the TRANSACT and APPLCTN information to your IMS stage I system definition and appending the TRXTRLR value to each.

Additional Requirements for Multiple ADFIDs

The following topics describe the conditions under which you are required to install another ADFID:

- You want the same version of IMSADF II but at different maintenance levels (in other words, for test and production systems).

- You want a test copy of IMSADF II dynamic rules databases that is in DB2 format.

Separate DB2 Tables

An exception occurs when you have selected DB2 as your IMSADF II database access method. Because DB2 names the tables specifically in the master Table Handler rules, you cannot have separate test tables. There are two master Table Handler rules per ADFID, called ????MSTR and ????SPRS. The tables reside in the DB2 address space, so you must install another ADFID to have a separate set of tables for testing.

Installing for Another Subsystem

You must install another ADFID for a second subsystem (IMS). The assumption is that you have at least one ADFID installed on this release of IMSADF II.

You may or may not elect to share SMP data sets among ADFIDs.

If you do share SMP data sets, see “Sharing SMP Data Sets Among Multiple ADFIDs” on page 86 for a checklist to follow for installation.

If you want the other subsystem to be separate, see “Multiple IMSADF II Systems on Different SMP Data Sets” on page 86.

SMP Considerations for the Multiple ADFID Environment

This chapter has described the multiple ADFID environment from the subsystem and IMSADF II standpoints. If you have decided that another ADFID is necessary, you can elect to use separate SMP data sets for each ADFID or to share SMP data sets among all ADFIDs. This section is only concerned with SMP requirements for installation and IMSADF II maintenance. SMP cannot maintain multiple releases of a product on the same data sets, so this description is limited to ADFIDs at the same IMSADF II release level.

You can install a second ADFID the same way you installed the first: by doing a complete installation using different SMP data sets. If you want to apply maintenance to SMP data sets separately, this is what you must do.

If you want to maintain SMP data sets at the same level and are willing to follow a few other restrictions, you can use the same SMP data sets, as well as common IMSADF II libraries, for all ADFIDs. This approach has several advantages:

- You save space by allocating only one set of SMP data sets.
- You save space by allocating only one set of IMSADF II common libraries (those with a high-level qualifier of ADFNODE).
- You save installation time by not having to ACCEPT the FUNCTION SYSMODS multiple times.
- You save maintenance time because one APPLY puts the same service on **all** ADFIDs at once, keeping them synchronized.
- You save time if you have to add another ADFID in the future. This is because you do not have to ACCEPT the FUNCTION SYSMODS, then put on all main-

tenance up to the level of your original ADFID **before** you begin to install the additional ADFID.

Both options are described in the following sections; the choice is yours.

Multiple IMSADF II Systems on Different SMP Data Sets

You may decide to have separate IMSADF II systems with different maintenance levels. Both systems can execute under the same subsystem, but they must be put on different SMP data sets so maintenance can be done separately.

You must do a complete reinstallation of IMSADF II, allocating separate SMP data sets. The only difference in installation occurs because you do not RECEIVE/APPLY/ACCEPT the IADF feature (JGE2210) on a second or subsequent ADFID.

Sharing SMP Data Sets Among Multiple ADFIDs

Use these instructions if you want to maintain multiple ADFIDs at the same level. In this situation, all ADFIDs share the same common libraries (those using high-level qualifier ADFNODE) and SMP data sets.

All ADFIDs must share the **same** ADFLOAD library, which is the SYSLIB for IMSADF II. The naming convention for IMSADF II load modules (those created by job ADFLINK) results in different names based on the ADFID selected. Many IMSADF II MOD entries have LMOD subentries, which have unique names (prefixed by ADFID). If you APPLY service and SMP data sets are shared among ADFIDs, SMP links an affected module to all load modules named in the LMOD subentry of the MOD statement. The SYSLIB must contain all LMODS referenced, or the APPLY will fail.

If you want to establish multiple ADFIDs on the same SMP data sets and IMSADF II common libraries, your installation procedure is altered somewhat from the one you followed when you installed the first ADFID. A checklist of the modified process for an additional ADFID follows:

1. Create a separate IMSADF.INSTALL by copying your existing one.
2. Run job ADFAALOC, but:
   a. Allocate only the data sets unique to an ADFID, designated by the high-level qualifier NEWADF (these data sets are also described in “Unique ADFID Libraries” on page 2).
   b. Expand the current allocation of IMSADF.ADFLOAD by the same amount as is specified in this job, since the distributed ADFLOAD allocation is sufficient for one ADFID.
3. Expand your CDS/target CSI, since you will be adding additional LMOD subentries to each affected MOD. Other SMP data sets should be unaffected.
4. Bypass this step if you are using the same subsystem as before, and go to step 5.

   Otherwise, install the second subsystem feature (if you already have IMS installed), if that was your reason for the additional ADFID.

   a. Add any required data set names for the feature to the ADFSMP4 procedure or to DDDEFS.

b. Run ASMPIMS (SMP/E users must go through the ACCEPT for the feature, using its dialogs).

5. Add the new options to the ADFOPTNS job.
   - You must keep the same ADFNODE as the original one, since you are sharing those data sets.
   - Enter a unique TRXTRLR value, since the transaction drivers of all supplied SYSIDs reside in the same ADFLOAD library.
6. Run the jobs described in Chapter 5, “Running the Customized Installation Jobs” for your environment.

There are some additional considerations. If you decide to share SMP data sets among ADFIDs, it is **strongly recommended** that you allow IMSADF II to have its own SMP data sets and not share them with IMS or other products. Doing this should eliminate the need to ever rebuild zones. If you are doing this, you can skip the rest of this chapter.

Sharing SMP Data Sets with Other Products: The following information applies only if you share SMP data sets with other products:

- If you have to unload/reload your CDS/target zone, run the ADFUNLOD/ADFRELOD jobs from IMSADF.JCLLIB for **each** ADFID.
- This consideration involves the GENERATE command; GENERATE applies only to SMP/E and is another way to build a new target zone and libraries by reading the old target zone and creating a job stream.

  IMSADF.RULLIB is a library that is unique to an ADFID. Members are created by the Rule Generator, a program not supported by SMP. You may or may not elect to place your application rules in this library. From an installation point of view, RULLIB is only included in the DDDEF, since certain link-edits name members of this library using an INCLUDE statement. All rules named in INCLUDE statements in job ADFLINK have the ADFID prefixed to them and are unique, regardless of environment. (The sample application SAMP, which is not involved in SMP entries, contains the same member names regardless of the ADFID.)

  The GENERATE command builds jobs, one of which is a link-edit. It constructs job control statements using DDDEF information as well as JCLIN contained in the target zone. The RULLIB DD statement output by GENERATE uses the ddname specified in your DDDEF (that of your original ADFID). When the job runs, any other ADFIDs will not link correctly since their RULLIBS are not present.

  Before submitting the created jobs, add all other RULLIB DD statements for other ADFIDs to the RULLIB concatenation, so that the desired rules are included correctly.

Chapter 8. Performance Considerations

This chapter addresses performance issues for subsystem environments.

Regardless of your subsystem, you can reduce paging when you put primary online execution time module MFC1V20 in the link pack area (LPA). Since MFC1V20 is reentrant, you use one copy of it no matter how many regions you have.

Note: MFC1V20 has no subsystem-dependent code, so it can be used for IMS.

IMSADF II application system performance is affected by the same considerations as any other IMS application program, such as choice of access method, database size, buffer pool size, and transaction volume. Tuning of the system, if necessary, should be approached as for other IMS applications. General IMS tuning considerations are discussed in the IMS library.

IMS Message Region JCL

To reduce directory searches, IMSADF II recognizes two libraries that contain rules. Specifying `RULESDD=`*`ddname1`* and `COMPMDD=`*`ddname2`* on the DEFADF macro indicates that there will be DD statements in the message region JCL, where certain types of rules can be found. For maximum performance, specify both parameters (if you use composites). If you specify both, you must place composite rules load modules in the library whose ddname is the COMPMDD value, and IMSADF II will search for the rules load modules there. Other rules must be placed in and searched for in the RULESDD library only. Companion parameters let you specify an optional number of dynamic BLDL entries that are managed for each library on a frequency-of-use basis; this is similar to the number of entries maintained for the IMS message region.

If these RULESDD and COMPDD options are specified and the DD statements are either missing from the message region JCL or specified as `DUMMY`, then STEPLIB is searched for rules, depending on what you specified for the RULESDD option. The search path length causes performance degradation.

The following example shows the library portions of the JCL for bringing up the IMS online message processing program (MPP) region:

```
//STEPLIB    DD  DSN=IMSADF.RULLIB,...
              DD  DSN=IMSADF.ADFLOAD,...
              DD  DSN=IMSVS.RESLIB,...
//ddname1    DD  DSN=IMSADF.RULLIB,...
//ddname2    DD  DSN=IMSADF.COMPLIB,...
//ADFDUMP    DD  SYSOUT=A
```

Note: The Rule Generator normally puts the transaction driver link-edits into the rules library (IMSADF.RULLIB), but you should override this action by directing these modules to `LINKLIB=`*`ddname`* on the GENERATE statement. The driver link-edit contains the IMS transaction name that the region controller must load. If you put the transaction driver in IMSADF.ADFLOAD, then IMSADF.RULLIB should be removed from the STEPLIB concatenation.

A technique you can use to reduce directory searches is to create several rules libraries concatenated under the RULESDD name. These libraries, which you must control, could only have rules beginning with certain letters of the alphabet (for

example, three libraries consisting of A-I rules, J-R rules, and S-Z rules). Put the libraries in the concatenation in reverse collating sequence.

Programs required to order and control such libraries must be written by you.

The ADFDUMP DD statement is necessary when the IMSADF II trace facility is being used. Tracing should only occur when IBM support personnel are tracking problem situations. For more information on this feature, see *IMS Application Development Facility II Version 2 Release 2 Diagnosis Guide*.

Note: Since tracing has associated overhead, you want to run in production with all tracing off, unless you encounter a specific problem.

Transaction Scheduling

IMS provides performance features in the area of scheduling; the features include parallel DL/I and wait-for-input. These performance features are specified in the APPLCTN and TRANSACT macros and should be considered for transactions that are used extensively. Conversational standard or special processing transactions, in which one IMS transaction provides several IMSADF II selections, are prime candidates for the parallel DL/I or wait-for-input features. See the IMS library for information on how to specify these performance options in the TRANSACT and APPLCTN macros. When you are in a storage-constrained environment, it is advantageous to specify IMSADF II transactions as high priority in a limited number of regions (instead of low priority applications in many message processing regions).

You should define IMS transaction cluster codes in Secondary Option Menu Rules to organize standard or special processing by database access method. IMS builds intent lists of database access for each application program to be scheduled. By organizing standard processing PSBs by database access, scheduling conflicts can be minimized and scheduling can be more efficient.

Rules Access

The order of search for a static rule is:

1. The IMSADF II preload list
2. RULESVS storage
3. The RULESBL/COMPMBL list for stored BLDL entries
4. If the rule is not in storage, a BLDL (stored if a value is specified for RULESBL/COMPMBL) followed by a LOAD (done as required)

Note: All of these supervisor services are charged to the IMS DC monitor's transaction CPU time, since they occur after the first DL/I call.

Using the IMSADF II Preload Function

IMSADF II's own preload function should be used to preload static rules. Using the Rule Generator, you name the static rules to preload, and the Rule Generator produces load module ????PRLD, which contains a preload list. When the first IMSADF II transaction is scheduled into an MPP, BMP, or batch region, ????PRLD and all of the rules in its list are conditionally loaded. If you need a different preload list for each region because of activity, you must have a different ????PRLD module for each MPP or BMP and control this through STEPLIB. A more effective way to accomplish the same thing is to preload any rules that must

be available, regardless of transaction usage, and use RULESVS to manage the other rules (see “Managing Rules Storage” on page 90).

Preloading is not done unless you specify `V50RES=YES` on the installation panel. When an abend or pseudo abend occurs in the MPP, program control is detached and everything is reloaded when the IMSADF II transaction is scheduled in the region. This is the same processing that happens to any modules preloaded by IMS.

Managing Rules Storage

DEFADF parameter RULESVS manages rules storage dynamically for rules that are not preloaded by IMSADF II. In each message region, a table is maintained of rules currently loaded, by type of rule. When a requested rule is not obtained through the IMSADF II preload list, the table is scanned by rule type.

RULESVS storage is not obtained through GETMAIN, but IMSADF II keeps track of the amount of storage used by currently loaded rules. When the specified amount is exceeded, the least-recently referenced rules (of any type) are deleted until the threshold is met. In this way, the most active rules, whether individual or composite, are maintained in storage for immediate access.

The following rules must be loaded below the 16 megabyte line:

Segment Handler
Table Handler
Primary Option Menu
Secondary Option Menu
Composite rules load modules

All other rules load above the line. Rules loaded above the 16 megabyte line are not included in RULESVS storage and, in fact, are not deleted by IMSADF II. However, a list is maintained of **all** loaded rules by type, chained in frequency-of-use order, for fast access with minimum paging.

If you select RULESVS, you may want to eliminate your composite rules load modules (see “Composite Rules Load Modules” on page 91), as they use larger amounts of RULESVS storage than individual rules. The rules that comprise the composite rules load modules can be replicated, both in other composites and individually.

If a rule is required that is not in storage or preloaded, a scan of the BLDL list (if you specified RULESBL or COMPMBL) is made to determine if the BLDL should be issued. If not, the LOAD is executed.

Note: If you relink a given rule during online execution, you may have to stop and start the message region to obtain the relinked copy.

Rules generated using any previous release of IMSADF II must be converted before they can be loaded above the 16 megabyte line.

The following rules, when relinked, **load** above the 16 megabyte line:

- Segment Layout
- Table Layout
- Input Transaction
- Output Format

- Mapping
- Static Audit Load Modules

Do not convert production rules until you are fully converted to this release of IMSADF II. Rules that are converted cannot be used in any prior release of IMSADF II when they are in 31-bit addressing mode.

In your investigation of performance problems, verify that application rules have been properly converted.

Using the IMS Preload Facility

The preload facility of IMS can be used for IMSADF II transactions. When used for transactions, load operations to obtain the module are eliminated (along with possible I/O to fetch the module); this is because IMS has already loaded the module and knows its location in memory. BLDL operations can also be reduced using the IMS dynamic BLDL facility. You could consider two IMSADF II-supplied transactions for IMS preloading:

- ????TOM (conversational sign-on)
- ????T99 (conversational termination)

Other candidates are your commonly-used application transaction drivers.

You can use IMS preload for rules (one BLDL per preloaded rule) if you do the following:

- Use RULESBL.

  If there are no BLDL entries maintained, a BLDL is issued every time you request a rule; otherwise the BLDL is only done the first time you invoke it.

- Establish a RULESDD, but do not place it in your message region JCL.

  If you have a RULESDD present in your message region, the LOAD points to that DD statement instead of to the STEPLIB from which IMS loaded the rule.

  Note: Any rules not preloaded normally have a longer search path through STEPLIB than through RULESDD.

Since the IMS preload list is unavailable, IMSADF II does a BLDL for the requested rule the first time it is invoked. When the LOAD is issued based on the BLDL entry, no I/O takes place, since the rule is in storage through the IMS preload.

Composite Rules Load Modules

Composite rules load modules should be used primarily in an environment where there is insufficient storage to devote to RULESVS management (see “Managing Rules Storage” on page 90).

Using composite rules load modules is an option that trades ease of maintenance for the performance benefits obtained by minimizing rule load operations.

The load module should comprise all rules required for a particular transaction, with these exceptions:

- Primary Option Menu Rule (conversational transactions only)
- Secondary Option Menu Rule (conversational transactions only)
- Any IMSADF II preloaded rules

The purpose of composite rules load modules is to reduce the number of rule searches, BLDL operations, and loads required during execution of a transaction. When the terminal user enters the IMSADF II transaction, all rules necessary for processing that transaction are loaded as a single entity. Even though the load module containing all the rules is larger than the individual rules, the elapsed time to load this one large module is far less than the time required to load individual rules modules.

Note: The number of static rules required for transaction processing varies based on the application. For example, a transaction that accesses four database segments and a work area (called a pseudo segment) requires that ten rules be loaded.

If you do not use the RULESVS option, IMSADF II does not delete the current composite rules load module when the transaction terminates, but leaves it in virtual storage until the next IMSADF II transaction is invoked. At this time, IMSADF II checks to determine if the new transaction is the same as the one just executed. If so, the existing composite is reused. If the two transactions are different, IMSADF II deletes the last-used composite rules module and loads the required composite.

Note: Composite rules cannot be preloaded by IMSADF II; composite rules are managed in RULESVS storage, since they are loaded below the 16 megabyte line.

If a rule is changed and composite rules load modules are not in use, the rule may be generated; all transactions using that rule have immediate access to it. When you use composites, you must execute the Rule Generator again for all composites using that rule.

Conversational Composites: A composite rules load module for conversational transactions is created by the Rule Generator with a name of:

```
sspgtt##
```

where:

ss is the first two characters of the application system ID
pg is the project/group
tt is the transaction ID
is the final two positions

The processing of a composite load module is controlled by an indicator in each TRXID entry in the Secondary Option Menu Rule (SOM); the indicator specifies whether the transaction is generated with or without a composite load module. If the composite indicator is on, the composite name is built and searched; if it is off, a normal rule search occurs. The indicator is turned on or off through the Rule Generator option CSOM (GENERATE statement).

Note: Before IMSADF II Version 2, if you selected either `COMPMOD=YES` or `ALT`, a BLDL was issued for **every** transaction as a composite across the ADFID.

An actual composite rules load module is created by the GENERATE CTLE function of the Rule Generator. Additional information about generating a composite load module is in *IMS Application Development Facility II Version 2 Release 2 Application Development Reference* under GENERATE options CTLE and CSOM.

Nonconversational Composites: A composite rules load module for nonconversational transactions is created by the Rule Generator with a name of:

```
ss#Ttt##
```

where:

ss is the first two characters of the application system ID
#T is constant
tt is the transaction ID
is the final two positions

If you decide to use composite rules load modules, define as many transactions as possible in composite form. A BLDL is done for each transaction and fails if there is no composite.

Note: Since there is no way to store this composite rule individually for nonconversational transactions, composites continue to be obtained across the ADFID when `COMPMOD=YES` or `ALT`.

Rules Access Summary and Recommendations

Assuming your virtual storage is sufficient, the best performance is obtained as summarized below:

1. RULESVS storage management

   Use IMSADF II preload only for those rules that must be quickly available, regardless of activity, and let IMSADF II manage individual rules through RULESVS. In that way, as your workload changes during the course of the day, your response time will remain short for the most heavily-used transactions. You may decide to eliminate composite rules load modules if RULESVS storage is sufficient for your needs. The advantages to doing this are:

   a. Composite rules, since they can contain Segment and DB2 Table Handler rules, must load below the 16 megabyte line, where storage may be constrained. When the composite rules are eliminated, all rules that can load above the line (where there are no RULESVS restrictions) will do so, freeing more storage below the line.

   b. When transactions are active, all their rules tend to remain in storage, thereby eliminating the reason for composite rules load modules (one load per transaction).

   c. Composite rules load modules can be quite large, and individual rules may be contained in many composites. When composites are managed by RULESVS, extra storage is occupied for each copy of a static rule that is imbedded in the composites. Elimination of composite rules makes the most efficient use of RULESVS storage, as well as reducing the maintenance required for related composites when a rule contained in them changes.

2. IMSADF II preload

   Preload only those rules that must have high availability, regardless of activity. In other words, preload only rules for transactions that must have fast response, even though they are seldom invoked.

3. IMS preload

   Preload ????TOM, ????T99, and your high availability transaction drivers.

4. Composite Rules

   With effective use of item 1, you can eliminate composite rules.

Virtual Fetch is not suitable for rules, since rules have no executable code.

If you are loading rules frequently and you place them on high speed devices without contention, performance improves without changing the application.

Work Database

This topic applies only to the DL/I Work database. Many of the listed tuning considerations apply to all DL/I databases, such as block sizes, data set placement, and buffer guidelines. The only option specific to IMSADF II is installation parameter WRKREDU. Because the Work database is frequently accessed during IMSADF II processing, “Root Addressable Area and Root Anchor Points (RAA and RAPs)” on page 95 makes recommendations based on the database key.

The SPA/Work database combination is the recommended choice in the IMS DL/I environment. It requires less virtual storage in the message queue pool than a large SPA, and if the specified block size is large enough to contain all Work database segments required to process an IMSADF II transaction, the number of I/Os necessary to process the transaction can be reduced to two. For more information on the considerations involved in this choice, see “IMS SPA and Work Database Considerations” on page 13.

Block Size

To reduce I/O to the Work database (since most transaction response time is waiting on I/O completion), the block size should be defined so that all segments required by a transaction are contained in one physical block. This can be accomplished by setting the block size of the Work database to a size that contains the total number of segments required by most IMSADF II transactions.

The most common block size used to give best performance is 32K. This block size fits both into an OSAM 32K database buffer subpool and also on a 3380 or 3375 track. This block size wastes space on the 3380 DASD device; however, the allocation of the data set is relatively small and significant benefit can be obtained by this choice.

Buffer Pool Size

If the access method is OSAM, isolate the database to its own subpool. The buffer pool subpool size for the Work database should be 32K or the closest larger value to your WRKAREA size. The important point to remember is that you want an entire Work database record contained in one buffer.

Note: This objective may be impossible to achieve when WRKAREA exceeds 32K (see “DEFADF Macro (Defining Basic IMSADF II Characteristics)” on page 15 for the current maximum).

The minimum number of buffers specified for this subpool should equal the number of message regions that can execute IMSADF II transactions; this is to allow a maximum of two I/Os to this database per transaction:

- One to read the database at the start of the transaction
- One to write the changes back to the database transaction at termination

Allocating additional buffers to this subpool can help. It may be possible to reuse the database record in the subpool, thereby eliminating I/O to reread the record. This occurs when the same logical terminal user enters another transaction before the Work database record is flushed from the database buffer subpool.

Data Set Placement

The IMSADF II Work database should be allocated on a lightly-used volume. If the I/O rate to this data set becomes such that I/O response time is outside device recommendations, the database should be split on two or more volumes. Splitting allows the I/O rate to each volume to be reduced, and the I/O response time should also be reduced.

Root Addressable Area and Root Anchor Points (RAA and RAPs)

The size of the Root Addressable Area (RAA) and number of Root Anchor Points (RAPs) is critical to the performance of the IMSADF II Work database from the standpoint of Program Isolation (PI) contention. PI waits (and even deadlocks) can be experienced on this database if there are insufficient RAPs, since the PI enqueue is on the RAP and not on the individual record. You can substitute a user-written sequential randomizer for this database so that when the key of the database (LTERM name) is randomized, there are very few, if any, synonyms.

The database should be allocated with one RAP per block, and the size of the RAA should be large enough so that the possibility of randomizing to the same block for two or more LTERM names is at a minimum.

Run the DBT HD Tuning Aid utility (5668-856) to analyze the RAA and number of RAPs, to determine what combination gives the best allocation and performance for this database.

Reduction/Compression

The installation WRKREDU option compresses the DL/I Work database before updating it at the end of a conversational iteration. Reduction works on Segment Layout and Input Transaction Rules only. Other areas remain intact. When the Work database is read again, it is expanded and the affected rules are reloaded.

The Work database area is reduced in main storage to determine if segments can be eliminated; if, because of the transaction content, none can, the reduced area is not chosen and the intact work area is written. The overhead involved is the time it takes to reduce the area contents.

Note: If reduction is not advantageous for a transaction, the work area is **not** reduced on subsequent transaction iterations in the same message region.

The primary advantage to using Work database reduction is that smaller change records are written to the IMS log, thereby reducing the overall volume of data. I/O to the database may also be reduced, since fewer segments are written.

Work database reduction is a part of the IMSADF II product. Since reduction is **not** an Edit/Compression routine, you can write your own edit routine to compress the rest of the Work Area: the User Communication Area and data beyond the Input Transaction and Segment Layout Rules. The fixed portion of the work area (800

bytes) is not compressed. “SPA Work Area Reduction Calculations” on page 184 gives you formulas to determine how much reduction you can achieve for your transactions.

Based on the transaction being processed, WRKREDU may or may not reduce the number of segments; therefore, you cannot count on any reduction in the PSB Work Pool (PSBW) in the IMS control region. Additional main storage, in the amount that was obtained for WRKAREA, is obtained via a GETMAIN; be sure to add this amount to your region requirements (see “IMS Storage Requirements” on page 5).

To achieve reduction of at least one segment requires:

1. Additional storage, equal to the size of your Work database record
2. CPU time to do the compression
3. Reload of the eliminated rules on the next conversational iteration

   Note: Unless you preload rules or use composite rules load modules, do not consider the third option from a response standpoint.

The WRKREDU option is **not recommended** for transaction response objectives.

Note: On output, IMSADF II always writes only the number of segments used during a conversational iteration, regardless of the maximum number of segments defined in the DBD.

Chapter 9. National Language Support

This chapter describes:

- General components of IMSADF II national language support (NLS)
- How to install a primary, alternate, and user language
- Replacing and adding the DBA transaction

Overview

IMSADF II supports application definition using many languages. IMSADF II supplies the following languages: English, French, German, Korean, Japanese, Portuguese, Spanish, and Swedish. IMSADF II also provides language components based on English, which you can modify for language requirements beyond the supplied languages. Designation of a language is indicated by a single numeric character suffix on each applicable product element. NLS is visible to the terminal end user via screens and messages, but is not intended to cover all areas in which the application developer is involved and the end user is not.

At installation time, you specify a default language and a set of alternate languages for each ADFID. The alternate languages are associated with SYSIDs under the ADFID when rules and screens are built:

- With the Rule Generator, and
- At execution time when the Lockword exit is invoked.

The Lockword exit is a key component in multilingual processing. IMSADF II support allows the same transaction code to operate across SYSIDs, allowing end users to see their transactions in the language of their choice.

Support by Component

The areas of IMSADF II in which NLS may be a consideration are described in the following subsections.

Rule Generator

The Rule Generator builds MFS source literals in the language defined in IMSADF II options; the Rule Generator does this using the unique, language-specific module MFC1G40. Language translations (one per language) are selected based on the default installation language and specifications on the SYSTEM statement. Additional keywords supplied to help NLS in the Rule Generator are listed below. The *IMS Application Development Facility II Version 2 Release 2 Application Development Reference* describes each keyword in detail.

- USRLANG on SYSTEM and GENERATE statements
- ALIAS statements, which let you generate a set of rules that can be used across multiple SYSIDs. This is how you generate an application suitable for multiple languages.

Rule Generator keywords and error messages remain in English.

Execution Time Considerations

Execution time IMSADF II diagnostic messages are generated from the appropriate translation module (MFC1V36n). Other messages and screen values are your responsibility. You must maintain user messages in the appropriate Message database under the corresponding SYSID.

For multilingual support, you must write a Lockword exit.

SYSIDs are not checked in any IMSADF II internal field processing, allowing Segment Layout rules, Mapping rules, and Output Format rules with the same content to be used across languages (SYSIDs) for the same transactions. SYSID is still used as part of the name of the rules, so only the appropriate rules are accessed.

Batch Processing

IMSADF II batch processing uses the same philosophy for language support as the online routines that control the language used in the printouts. However, because of the possibility of multiple sign-ons in a batch run, all SYSIDs with a common language should be processed in a single run and kept separate from SYSIDs for other languages. The batch driver runs with multiple languages, but the printed output will contain a mixture of languages if SYSIDs for different languages are processed in the same batch run.

No Modifications

The following components are not designated for the terminal end user and, therefore, have no special modifications made to them.

- IMSADF II Supplied Applications. The DBA and SAMP applications are supplied in English. The SAMP application validates the installation of IMSADF II. The DBA application (SYSID is the same as ADFID) is a support tool for use by data base administration or the application developer and is also not seen by the end user. If you determine that these applications are to be implemented in a language other than English, see “Replacing/Adding the DBA Transaction” on page 105 for instructions.
- Dictionary Extension. Extension data is organized by SYSID and can thus carry language-dependent data without changes.
- High-Level Audit Language. The High-Level Audit Language supports conditional expressions that test the language specified for the SYSID. As a result, audit subroutines have some sensitivity to the requirements of multilingual applications.
- Rules Documentation

Sign-on Processing and the Lockword Exit

IMSADF II support for multilingual applications depends on your implementation of a user Lockword exit (a standard part of IMSADF II). The Lockword exit defines a correspondence between a language and a SYSID. The support takes effect after the end user has entered identifying information on the sign-on screen. It is a good idea for the developer to define a multilingual sign-on screen through the Rule Generator screen image capability.

The sign-on module sets the SPAULANG field in the SPA with the default language code for the installation, and then calls the Lockword exit. Your code in the

Lockword exit module can override and set the SPAULANG and SPASYSID values, as appropriate, for the detected SYSID or the project group. Make sure that the proper language and SYSID are set. If they do not correspond, the screen text and application messages will not match. On return from the Lockword exit, the sign-on modules validate the SPAULANG value against the languages defined at installation time in the USRLANG and ALTLANG keywords of DEFADF.

Note: If English is an alternate language, the SPAULANG value is stored internally as a blank. Be sure to use a blank in your Lockword exit logic and not E.

The exit sets the common screen prefix in SPAMFSPF, which is either the first two characters of the ADFID or SYSID named for the corresponding language. If the SPAULANG value is invalid, the prefix used is that of the ADFID. Selection of the eight common IMSADF II screens is done using the SPAMFSPF value set at sign-on.

The SYSID check is made after return from the Lockword exit. Though this is not strictly an NLS function, if the SYSID field in the project group segment is blank in your exit, no SYSID check is done. This lets SYSID checking be bypassed under control of the Lockword exit or without intervention of the Lockword exit for those installations where this is necessary.

Note: For multilingual applications, application developers should evaluate the F option on the Primary Option Menu (project group switch). A project group switch does not call the Lockword exit routine and so cannot cause a language change.

Product Elements Supplied

The following elements are distributed with the appropriate suffix for supported languages:

MFC1G40n module--Rule Generator IMS screen source template
MFC1V36n module--IMSADF II messages and decimal separator
ADFMFSLn source--IMS base screen source input
TRANSINn source--batch driver input (cumulative)
TRANSUPn source--batch driver input (since last IMSADF II release)

The English language modules are link-edited with their components as part of the install process. Other language modules are loaded as required.

Module MFC1V36n tells IMSADF II what to use for the decimal separator. The DECSEP field in module MFC1V36 can be changed so that a comma or period is used.

You are supplied one set of English source for each of the listed elements to help with the user-defined language sets.

Note: Installation job ADFULNG installs the above components for user languages.

For user languages, examine service for the English language version of the elements and make corresponding modifications via usermod in the selected language.

Developing Multilingual Applications

To run multilingual applications, the application developers must do the following:

- Install primary, alternate, and user languages, as described later in this chapter.
- Define base screens for the required alternate languages, based on your subsystem. Execute installation jobs SCREENS and MFSJOB.
- Specify the languages and associated SYSIDs with the ALTLANG keyword, and run job ADFOPT (or ADFUSER if you have already installed IMSADF II).
- Code the Lockword exit to determine and set the SPAULANG, and file SPASYSID values as appropriate. Install the Lockword exit using “Installing Sign-On and Sign-Off Exits” on page 77.
- Generate corresponding Input Transaction Rules and screens for the transactions under each of the SYSIDs.
- Provide end users with the sign-on information that gives them access to the transactions for their preferred languages.

Common Tasks Performed in NLS Implementation

The rest of this chapter is the detailed instructions you must follow to achieve your language objectives. To keep the instructions uncluttered, references from specific lists are made to tasks that are described in this section. Based on your subsystem environment and access method for the IMSADF II dynamic rules databases, some job names and procedures may change, though the function is equivalent.

Certain tasks described in this section have different job names, based on the online environment you have selected, but the function is identical . You invoke the following based on your environment:

- ADFMFS
  This IMSADF II macro creates source for the supplied screens. Only NLS-related keywords are mentioned in this chapter.
- SCREENS
  This job executes from IMSADF.JCLLIB to invoke the language conversion process for the supplied IMSADF II screens.
- MFSJOB
  This job executes from IMSADF.JCLLIB to put MFS formats for supplied IMSADF II screens in online subsystem libraries.

Note: You can generate a significant number of screen formats, since they are expanded by the number of languages supported.

Base Rule Generation

Some base rules may be regenerated; their names change based on the database access method:

- RLGAUDIT (DL/I) = DB2AUDIT (DB2)
- RLGSIGNP (DL/I) = DB2SIGNP (DB2)
- RLGMSGS (DL/I) = DB2MSGS (DB2)
- RGLBASE (all)
- RGLSAMP (all)

DB2 Input to the Batch Driver

Batch driver input supplied with IMSADF II is in IMS format. Convert it to DB2 format by invoking ADFTRTR, which uses TRANSINn(UPn) as input and places output in IMSADF.JCLLIB as TRDB2INn(UPn).

NLS Customization

When you change existing options while installing a language, you invoke the CHANGE option of the ISPF Installation Dialogs to customize job ADFUSER. Subsequently, select the CUSTOMIZE option and recustomize the following members selectively from IMSADF.JCLLIB:

SCREENS
MFSJOBS
TRANSINn/TRANSUPn
DBUPDATS
ADFDATAS
PROCBS
ADFTRTR (if DB2)

Installing A Primary Language

The flow for installation of a non-English language parallels that for normal installation; however, some steps need to be explained. The following example helps illustrate NLS implementation.

```
DEFADF  USRLANG=G,ALTLANG=(J,KOBE,DBCS)
```

In this example, German is the primary language and German messages are listed under the ADFID. Japanese is the only alternate; Japanese system messages are found under the SYSID of KOBE; IMSADF II-supplied screens have the prefix `KO`.

1. Macro ADFMFS contains a `USRLANG=` keyword (`G`) entered on the ISPF Installation Dialogs panels. Notice that there is no place for ALTLANG and its DEFADF values. You must generate base screens for any alternate languages separately. See “Installing An Alternate Language” on page 102 for detailed information on how to do this.
2. During job ADFLINK, the Rule Generator is invoked for the pertinent online supplied rules (see “Base Rule Generation” on page 100).
3. Notice that jobs ADFDATA and DBUPDATE are customized for TRANSING. All IMSADF II supplied audits, profiles, and messages are loaded in German. The messages for SAMP, even though it is English, are present in every TRANSINn supplied. If you require DB2 input, see “DB2 Input to the Batch Driver.”
4. When you run the Installation Verification Procedure (IVP) after installation, the messages for your ADFID are presented in German; the screens for SAMP appear in English.

Now you have an IMSADF II installed with German as the primary language. See the next section for information on how to install Japanese as your alternate language.

Installing An Alternate Language

If you want to install an alternate language, it must be done after initial installation. Language options are set either at initial installation by job ADFOPT, or at post-installation by invocation of the DEFADF macro in job ADFUSER. In the previous section, an example was presented that will also be used in this description. The pertinent DEFADF keywords in the example were `USRLANG=G` and `ALTLANG=(J,KOBE,DBCS)`, which specifies that German is the primary language and Japanese is the only alternate language.

Note: An editor is helpful in implementing changes.

1. Run job ADFUSER after changing options, as described in "NLS Customization" on page 101. If your alternate languages were established at initial installation, this step is unnecessary.
2. To generate base screens for IMS, run job SCREENS, changing the SYSPUNCH member BASEMFS to BASEMFSJ. Then run job MFSJOB to create the screens, using BASEMFSJ as the only SYSIN.
3. Though your TRANSINJ was customized, you must change it if you want to use it as an additional language. These changes are easier to make if you copy the TRANSIN for your language from IMSADF.ADFSLIB.

   Note: The *IMS Application Development Facility II Version 2 Release 2 Application Development Reference* contains detailed information on batch input and formats for the IMSADF II dynamic rules databases.

   a. Delete the profiles, or you will get IMS DL/I status code `II` when you insert the project group (PG) segment.
   b. Delete the audits, since they will be unused. This leaves only the messages.
   c. Change the `????B` character string to the ADFID plus the letter B (in other words, `????B` becomes `ADF2B` if `ADF2` is your ADFID) in all places.
   d. Change the remaining `????` values to your SYSID in DEFADF ALTLANG.
   e. Change the two question marks to the first two characters of your SYSID value in DEFADF ALTLANG.
   f. If you did not use the DELIM default ($$), change $$ to your DEFADF DELIM values.

   Run job DBUPDATE changing SYSIN to the TRANSIN for your language (TRANSINJ). See "DB2 Input to the Batch Driver" on page 101 if you require DB2 input. Your dynamic rules databases are updated with the messages for your language.

The alternate language is now installed. Go through this sequence once for each alternate language in DEFADF.

Installing User Languages

You can install a user language during or after initial installation. Language options are set at initial installation by job ADFOPT or at post-installation by invocation of the DEFADF macro in job ADFUSER.

The components required for installation are:

- Rule Generator screen templates (MFC1G40n/MFC1G60n)
- Execution time internal messages (MFC1V36n)
- System messages, audits, and profiles (found as TRANSINn or TRANSUPn)
- Inner macro ADFMFSLn (invoked to generate the base screens in your language)

The `n` shown above is a suffix you choose to designate the desired language; `n` must be the same as what you entered on the ISPF Installation Dialogs panels for keywords USRLANG or ALTLANG.

The listed items are put on your system in the form of an SMP usermod by job ADFULNG. Copy the above members in source form from the English version of the appropriate distribution libraries. Do the translations and customization that is required (usually ADFID/SYSID). RECEIVE and APPLY the usermod, which adds the appropriate members to your IMSADF II system.

If you want supplied transactions in your language, translate them from English and invoke the Rule Generator, though this step is not recommended. See "Replacing/Adding the DBA Transaction" on page 105 for instructions about this.

Process for Installing a Language

The steps you follow to install a language vary based on your purpose and when you implement your language. Four different situations are considered in the sections that follow.

Note: An editor is helpful in implementing your changes.

Post-Installation Alternate Language

The process for installing an alternate language after installation is as follows:

1. Run job ADFULNG to put the language modules on your system.
2. Run job ADFUSER after changing the options, as described in "NLS Customization" on page 101. If your alternate languages were established at initial installation, this step is unnecessary.
3. To generate a base screen, in job SCREENS code `USRLANG=n` and `SYSID=your SYSID` in the DEFADF ALTLANG keyword.

   When you execute job SCREENS, change the SYSPUNCH member BASEMFS to BASEMFSn. Then run job MFSJOB to create the screens using BASEMFSn as the only SYSIN, replacing your original screens.
4. Copy IMSADF.ADFSLIB(TRANSINn) to IMSADF.JCLLIB(TRANSINn). The following steps update your databases with the messages for your language.

   Note: The *IMS Application Development Facility II Version 2 Release 2 Application Development Reference* contains detailed information on batch input and formats for the IMSADF II dynamic rules databases described below:

   a. Delete the profiles, or you will get IMS DL/I status code `II` when you insert the project group (PG) segment.

   b. Delete the audits, since they are unused. This leaves the messages only.

c. Change the ????B character string to the ADFID plus the letter B (in other words, ????B becomes ADF2B if ADF2 is your ADFID) in all places.

   Note: The sequence of this change and the next is order dependent.

d. Change the remaining ???? values to the SYSID in DEFADF ALTLANG.

e. Change the two question marks to the first two characters of your SYSID value in DEFADF ALTLANG.

f. If you did not use the DELIM default ($$), change the $$ to your DEFADF DELIM values.

Run job DBUPDATE changing SYSIN to your TRANSINn. See "DB2 Input to the Batch Driver" on page 101 if you require DB2 input.

Post-Installation Primary Language

The process for installing a primary language after installation is as follows:

1. Run job ADFULNG to put the language modules on your system.
2. Run job ADFUSER after changing the options, as described in "NLS Customization" on page 101. If your alternate languages were established at initial installation, this step is unnecessary.
3. To generate a base screen for IMS, run job SCREENS without changes. Then run job MFSJOB to create the screens using BASEMFS as the only SYSIN, replacing your original screens.
4. Copy IMSADF.ADFSLIB(TRANSINn) to IMSADF.JCLLIB(TRANSINn).

   Choose one of the following two methods for placing messages on your databases.

   Method 1

   To add a user language's messages, which REPLACE the ones originally installed (which use the ADFID as the key), follow this procedure.

   Note: The *IMS Application Development Facility II Version 2 Release 2 Application Development Reference* contains detailed information on batch input and formats for the IMSADF II dynamic rules databases described below.

   a. Delete the audits and profiles, since there is no language dependency and they were initially added.

   b. Change the ???? values to your ADFID.

   c. Change the two question marks to the first two characters of your ADFID.

   d. Change the batch input from insert mode (4) to replace mode (5), so that your input replaces the input initially added.

   e. If you did not use the DELIM default ($$), change $$ to your DEFADF DELIM values.

   Run job DBUPDATE, changing SYSIN to TRANSINn, which updates your existing database with your new language. See "DB2 Input to the Batch Driver" on page 101 if you require DB2 input.

   Method 2

   You can use new dynamic rules databases if you want DL/I support for them. If you want to do an initial load, instead of the procedure previously mentioned, do the following:

a. Scratch and reallocate your IMSADF II dynamic rules databases.

b. Change the ???? values to your ADFID for your TRANSINn.

c. Change the two question marks to the first two characters of your ADFID.

d. If you did not use the DELIM default ($$), change $$ to your DEFADF DELIM values.

Run job ADFDATA, changing SYSIN to TRANSINn, which loads all the dynamic rules databases with your new language.

Initial Installation Primary Language

The process for installing a primary language at installation time is as follows:

1. Run job ADFULNG to put the language modules on your system. This **must** be done during installation **before** you run job SCREENS, or the macro invocation will fail.
2. Copy IMSADF.ADFSLIB(TRANSINn) to IMSADF.JCLLIB(TRANSINn).
   a. Change the ???? values to your ADFID.
   b. Change the two question marks to the first two characters of your ADFID.
   c. If you did not use the DELIM default ($$), change $$ to your DEFADF DELIM values.
3. Do the rest of the installation, as described in Chapter 5, “Running the Customized Installation Jobs,” based on your environment.

Initial Installation Alternate Language

Follow the procedures listed for post-installation alternate language, as all alternate languages are installed separately after initial installation.

Replacing/Adding the DBA Transaction

There is usually no need for jobs DBA and SAMP to be used in languages other than English. The method described in this section applies only in the IMS DL/I environment. To have the DBA transaction in your language, you can either add it to the English version or replace the English version. It is easier to add the DBA function than to replace the existing one. This discussion applies to **any** supplied or user language. First install the language using any applicable method discussed in this chapter.

Replacing DBA Transaction

To replace the DBA transaction after installation, do the following:

1. Copy the following members from IMSADF.ADFSLIB to a separate library: RGLAUDIT, RGLBASE, RGLSIGNP, and RGLMSG. Translate these members.
2. Customize the members as follows:
   - Make ???? and ?? the four or first two characters, respectively, of the ADFID.
   - Change USRLANG= to your language.
   - Change DEVNAME, DEVTYPE, and DEVCHRS to your desired values, usually the same as you supplied in DEFADF.

3. Run the Rule Generator procedure for each SYSIN supplied. The rules replace those previously created by job ADFLINK. Run the MFS utility to place the formats in IMSVS.FORMAT.
4. Your IMSADF II dynamic rules databases require no change, since there are no language dependencies in either the profiles or audits and the messages for the ADFID were initially added.

Adding DBA Transaction

To add the DBA transaction after installation, do the following:

1. Copy your TRANSINn from ADFSLIB, or use your existing TRANSINn in IMSADF.JCLLIB. Edit TRANSINn as follows:
   a. `????` and `??` become the four and first two characters of the SYSID for your language.
   b. Change the profile information to another project group (PG segment), since the supplied project group was already loaded.
   c. Delete the audits, since they were initially added and there is no language dependency.
   d. Change the batch message input from insert mode (4) to replace mode (5), so that your input replaces the input initially added.

   Run job DBUPDATE, changing SYSIN to TRANSINn, which updates your existing database with your new profiles and messages.
2. Copy the following members from IMSADF.ADFSLIB to a separate library: RGLAUDIT, RGLBASE, RGLSIGNP, and RGLMSG. Translate these members.
3. Customize the members as follows:
   - `????` and `??` become the four or first two characters, respectively, of ADFID or SYSID, depending on whether your language is primary or alternate.
   - Change `USRLANG=` to your language.
   - Change DEVNAME, DEVTYPE, and DEVCHRS to your desired values, usually the same as you supplied in DEFADF.
4. Run each member as SYSIN to your Rule Generator procedure. The rules add to those previously created by job ADFLINK. Run the MFS utility to place the formats in your subsystem library.

Chapter 10. Post-SMP Maintenance

IMSADF II is maintained through Program Update Tapes (PUT tapes). Changes to product elements appear regularly on PUT tapes. Changes are incorporated into your environment through SMP. Though base code is automatically relinked, there are additional steps you must take when the element that is changed is not a program load module. This chapter gives guidelines for accomplishing these steps. Job streams supplied at installation time can be modified for this purpose.

Maintenance to members in IMSADF.ADFSLIB can be supplied in either MACUPD (update) or MAC (replacement) form. Although SMP places the service on IMSADF.ADFSLIB, you may have to run additional jobs to make those changes an active part of your IMSADF II environment.

In Appendix F, “Source Members Requiring Special Handling” on page 188, there is a list of the non-macro members of IMSADF.ADFSLIB and the libraries where they are placed at the completion of installation. For SMP/E users, maintenance to these members is indicated by ++HOLD SYSTEM statements on the PTFs.

All jobs listed in this chapter are described in “Contents of IMSADF.JCLLIB.”

Installation Skeletons

Members of IMSADF.ADFSLIB that are in symbolic form require customization of one of the following libraries:

- IMSADF.JCLLIB
- IMSADF.DBDSRC
- IMSADF.PSBSRC
- IMSADF.RULES.SOURCE

To do customization, invoke the CUSTOMIZE option of the ISPF Installation Dialogs and select the affected member(s). Then do as follows:

- JCL members (JCLLIB)

  JCL is now ready to execute. If the change was made to a cataloged procedure, copy it to your PROCLIB. You can use job COPYPROC for this purpose.

- DBD members (DBDSRC) (DL/I environment only)

  Do a DBDGEN for the affected member of this library. Job ADFDBD can be used. If macro DBDW is changed by maintenance and you want to run ADFDBD after APPLY, concatenate the SMPMTS data set in front of IMSADF.ADFMAC for the SYSLIB DD statement of the step named WORK. For online execution, you must do an ACBGEN. You can use job ADFACB for this purpose, selecting affected members. If the change involves your application transactions, you must do a DBDGEN/ACBGEN for those also.

- PSB members (PSBSRC) (all environments)

  Do a PSBGEN for the affected member of this library. Job ADFPSB can be used. The SYSLIB DD statement for all the PSBGEN steps must have the SMPMTS data set concatenated before IMSADF.ADFMAC for the macros in APPLY status to be incorporated. For online execution, you must do an ACBGEN. You can use job ADFACB for this purpose, selecting affected

members. If the change involves your application transactions, you must do a PSBGEN/ACBGEN for those also.

- Supplied rules (RULES.SOURCE)

  Run the Rule Generator, selecting the affected member of IMSADF.RULES.SOURCE as input. Unless the affected member is a batch rule, run the MFS utility to place any format changes in your subsystem screen library. MFSJOB contains the steps required to do this for any supplied rule.

  If you use DB2 for the dynamic rules databases and any of the master Table Handler rules (DB2MSTR, DB2SPRS, and DB2STHR) are changed, run job ADF2PAL to precompile, assemble, and link their DBRMS (????MSTR, ????SPRS, and ????SPAW). Then rebind the supplied application plans found in member ADFTBLB, which is input to job ADFTBTSO.

- Dynamic rules databases

  Additions to the dynamic rules databases are found in IMSADF.JCLLIB(TRANSUP). Run job DBUPDATE to place the additions on your IMSADF II dynamic rules databases. If you use DB2 for your IMSADF II dynamic rules databases, convert the DL/I input to DB2 format using IMSADF.JCLLIB(ADFTRTR) first.

- For NLS users:

  1. If your language is a supplied primary language, run DBUPDATE, which is already customized correctly for your language.
  2. If you use a supplied alternate language, customize your TRANSUP as you did your TRANSIN, and run DBUPDATE specifying your TRANSUP as input.
  3. If you have another language, copy the English TRANSUP, translate it, and APPLY it as a usermod, superseding the one you specified in job ADFULNG. Customize TRANSUP in IMSADF.JCLLIB as you initially did your TRANSIN, and then run DBUPDATE with TRANSUP as input. (In the DB2 environment, convert TRANSUP after customizing it, using job ADFTRTR first.)

Rules Documentation System

The Rules Documentation system is controlled by a rule called ????BDDR (where ???? is the ADFID). ????BDDR was created by the Rule Generator during installation. It is composed of a large load module (DFRS720) and linked with special processing routines. A list of these routines is in “Members of ADFMAC (Special Processing Routines)” on page 191. You must rerun the Rule Generator for any IMSADF II transaction driver that includes a special processing routine at link edit time.

While SMP links maintenance to any of the modules that comprise the Rules Documentation system, the modules are not incorporated into your executing environment. You must rerun the Rule Generator when changes are made to the Rules Documentation system, specifying IMSADF.RULES.SOURCE (RGLBDDR) as input. Since this is a batch rule, there are no screen considerations. To determine the modules included in the driver link edit, look at at the output of the installation step in job ADFLINK, which produced it.

Maintenance may consist of program modules and special processing routines. See “Members of ADFMAC (Special Processing Routines)” on page 191 for steps you must take if the latter are changed. For SMP/E users, all maintenance has ++HOLD SYSTEM statements on the PTF.

PL/I Sign-On and Sign-off Exits

If you have sign-on (Lockword) or sign-off exits written in PL/I, you must relink them after maintenance is applied to either:

- ????TOM (sign-on)
- ????T99 (sign-off)

Note: If your exits are not in PL/I, they are not affected by maintenance; they are handled by SMP.

Transaction Driver Modules

Transaction driver link edits include IBM-supplied modules. If there is maintenance to these IBM-supplied modules, you must relink the affected transaction drivers by executing the Rule Generator.

User Language Modules

Any elements relating to installation of a user language are supplied in English only. Changes to any of the following modules are supplied in source form. You must examine the changes and APPLY them with another usermod to your user languages; this is done after translation.

MFC1G40
MFC1G60
MFC1V36
TRANSIN
TRANSUP
ADFMFSL

IMS Release Change

The ACDS corresponds to the distribution zone CSI, and the CDS is the target zone CSI.

Since the method of IMS installation requires that the ACDS be copied to the CDS, all other products also installed on those data sets are affected. The ACDS and CDS can be completely reinstalled, which may be desirable if they depend on IMS. If there are no dependencies, as is true with IMSADF II, the ACDS and CDS entries can be unloaded from the current IMS ACDS/CDS and reloaded to the new IMS SMP data sets only, or you can use the GENERATE command for SMP/E to reload the target zone CSI.

Two jobs are supplied for you in IMSADF.JCLLIB, one to unload (ADFUNLOD) and one to load (ADFRELOD). Both jobs use the supplied SMP procedure. If you have different IMS SMP data sets, you can change the symbolic information required for the jobs on the EXEC statement in the ADFRELOD job.

The ADFUNLOD job does not unload the SYS entries in either the ACDS or CDS, since it assumes you will be loading IMSADF II to the data sets on which your new release of IMS exists. If such is not the case (for example, if you created separate SMP data sets yourself), add the UCL UNLOAD statement for the SYS entries to the ADFUNLOD job stream so they reload successfully to empty ACDS/CDS. The ADFRELOD job adds the IMSADF II FMIDS to the PTS for you.

Release Change Considerations with Shared SMP Data Sets

Another situation can occur with SYSMOD entries if you share SMP data sets with IMS.

A superseded PTF that was never placed on the system is not selected when SYSMODS are UNLOADED FORFMID. If that PTF is a prerequisite for another on the new CDS, service does not apply, as shown in the following example.

1. PTF1 has no prerequisites and is being SUPerseded by PTF2.
2. You did not place PTF1 on your system, but you did APPLY/ ACCEPT PTF2. That placed an entry for PTF1 on your ACDS/CDS with the SUP notation only.
3. You now install a new release of IMS, and you choose to do the UNLOAD/RELOAD for IMSADF II.
4. Subsequently, you obtain the following: PTF3 PRE PTF1.
5. You cannot APPLY the service because you do not have PTF1 on your system.

What occurred was that you unloaded all entries for the IMSADF II FMIDS. A superseded PTF that is never placed on your system has no FMID information (it is a minimal entry); therefore, it was never unloaded.

Since most maintenance is for modules, the normal keyword specified on the PTF is SUP, and not PRE; thus this problem is unlikely to occur.

You have several choices if this problem does occur:

Choice 1. If you choose to deal with the problem after the fact and only if it is encountered:

- Use the BYPASS PRE option on the APPLY for PTF3, or
- Use the UCL feature of SMP to update the CDS/ACDS with the PTF1 SUPBY PTF2 entry.

Choice 2. At the time of the UNLOAD, you can:

- Find the IMSADF II FMID entries yourself and code them, adding them to the ADFUNLOD job, or
- Add UNLOAD SYSMOD SUP to ADFUNLOD, which unloads ALL superseded SYSMODS. In this situation, you can be sure you have all superseded SYSMODS.

Note: You do not have to be concerned with APARs, since a PTF does not precede or require an APAR.

When you have done either choice 1 or 2, PTF3 will correctly APPLY.

Choice 3. You can completely reinstall IMSADF II after you install the new release of IMS.

Note that the preceding discussion requires SMP expertise to evaluate.

If you are familiar with SMP and want to share SMP data sets with IMS, you will probably choose to deal with the problem after the fact and only if it occurs. If you do this, the BYPASS PRE option is the easiest to implement. This requires significantly less effort than reinstallation of the product.

Chapter 11. Sample Problem

This chapter presents the IMSADF II sample problem, which can be used to verify that IMSADF II installation was successful. The sample problem can also help you become familiar with operation of the system.

The sample problem uses the DL/I database DI21PART, which is provided with the IMS DB feature. You can use the sample problem if you have any of these installation environments:

- IMS environment with DL/I dynamic rules databases
- IMS environment with DB2 tables

This chapter is divided into the following major sections:

- Description of the sample problem
- Installation considerations
- Detailed, screen-by-screen discussion of the steps involved in using the sample problem

Sample Problem Description

The IMSADF II sample problem uses the DI21PART database provided with the IMS system. This database is a three-level, parts inventory data set.

The sample problem demonstrates how to use an IMSADF II system to locate segments and update, delete, or add data. These functions, which are handled by standard processing, are controlled by the rules furnished with the problem.

A special processing routine is also supplied to demonstrate how to use user-written procedural code. This special processing routine demonstrates interaction with the segment handler, data mapper, auditor, and SPA fields. A listing of this routine, which is written in PL/I and COBOL, is in Appendix B, “Sample Problem Examples.”

Five IMSADF II transactions are provided to access the sample database shown in Figure 45.

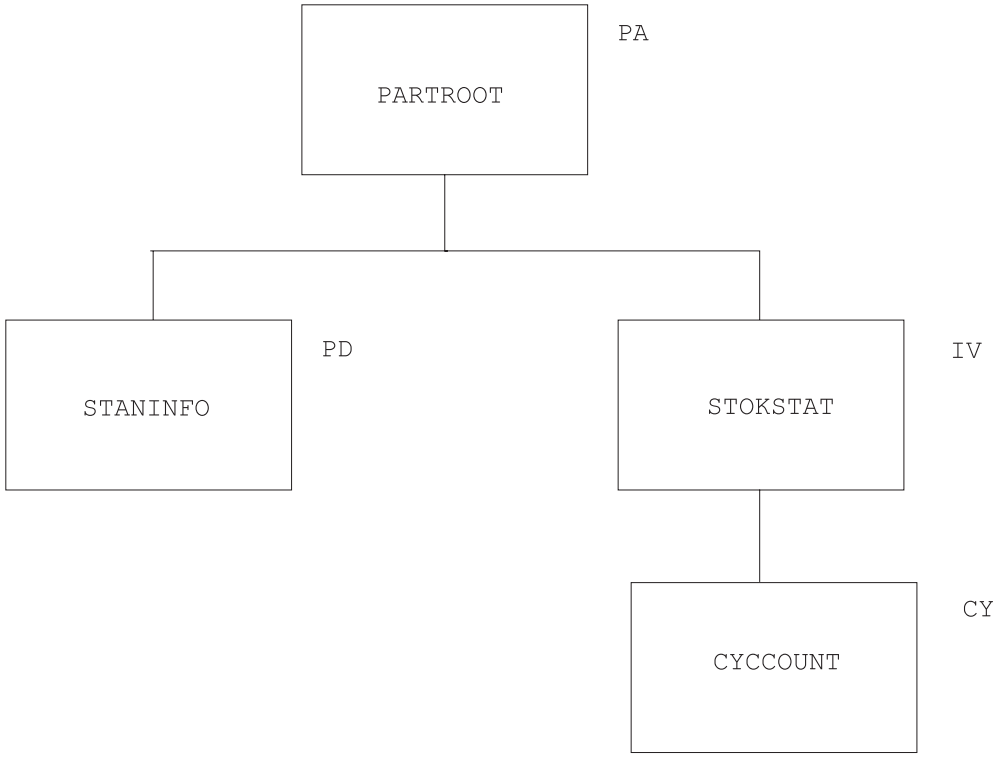

Figure 45. Sample Problem Application Database

The first four transactions are standard processing transactions used to access and update the following segments:

Transaction	Segment(s) Accessed
PA	Part Master
PD	Part Master, Standard Data
IV	Part Master, Stock Status in Inventory
CY	Part Master, Stock Status in Inventory, Cycle Count

The fifth transaction, CD, is used for special processing. The CD transaction lets you close an open order, increase the total stock quantity, or disburse a quantity from a given inventory location. The segments accessed are Part Master, Standard Data, and Stock Status. The Stock Status segment is updated using data entered at the display screen.

Installation Considerations

To run the sample problem, use the DI21PART database supplied with IMS. For information about this database, see the IMS library. Load the database and create the DBDs, as described in the IMS library.

All other installation requirements for the sample problem are handled through installation of the IMSADF II system. When installation is complete, the following sample problem components are available:

- IMS system definition input

  Appendix A, “Sample IMSGEN Control Statements,” provides the sample APPLCTN and TRANSACT definitions needed to define the sample problem to IMS.

  Members of IMSADF.JCLLIB, beginning with ADFC, contain the installation data required to do this.

- PSBs/ACBs

  The PSBs are supplied in IMSADF.PSBSRC and are generated by installation job ADFPSB. These PSBs are also included in sample problem ACBGEN as part of installation job ADFACB.

- Rules

  All rules necessary to run the sample problem are included in IMSADF.RULLIB. These rules are produced by the Rule Generator in job ADFLINK.

- Screens

  All required screen source is created by the Rule Generator during job ADFLINK and is placed in IMSADF.SCREENS. The screen source was also generated from the Rule Generator source. Job MFSJOB (IMS) put formats in the appropriate library.

- IMSADF II dynamic rules databases

  The Sign-On Profile database is loaded during job ADFDATA or DBUPDATE. This lets you sign on with **USERID=**999999, **Project=**Z, and **Group=**Z. In addition, system messages are loaded into the Message database, and audit descriptors for the sample problem are added to the Audit database.

Operating the Sample Problem

Once the IMS sample database has been loaded and the IMSADF II installation is complete, the sample problem is ready for execution. The rest of this chapter shows the screens that are displayed and the input required to run the sample problem.

The screens that follow are presented in pairs. The first screen in a pair shows the screen as it is displayed to you. The second screen shows your entries. The items that you enter are also shown right before the second screen, next to the word **ENTER**.

Note: If the sample problem database was modified by a previous run of the IMS sample problem, the values shown on the screens may not match those shown in the book. A reload of the database will solve this problem.

To invoke the sample problem sign-on screen, enter:

```
/FOR SAMP
```

Or if you are in the IMS environment, enter:

```
SAMP
```

```
                S A M P L E   P R O B L E M



        ENTER THE FOLLOWING SIGN-ON INFORMATION AND DEPRESS ENTER

                        -- USERID

                        -- PROJECT

                        -- GROUP

                        -- LOCKWORD


OPTIONALLY, ENTER TRANSACTION DETAILS FOR DIRECT DISPLAY
OPTION:     TRX:     KEY:
```

Figure 46. Sign-On Screen (Display)

```
     ENTER: 999999 -- USERID
                 Z -- PROJECT
                 Z -- GROUP
```

```
                S A M P L E   P R O B L E M



        ENTER THE FOLLOWING SIGN-ON INFORMATION AND DEPRESS ENTER

                  999999 -- USERID

                       Z -- PROJECT

                       Z -- GROUP

                         -- LOCKWORD


OPTIONALLY, ENTER TRANSACTION DETAILS FOR DIRECT DISPLAY
OPTION:     TRX:     KEY:
```

Figure 47. Sign-On Screen (As Entered)

```
                        P R I M A R Y   M E N U

OPTION:         TRANSACTION MODE:       IDENTIFIER:
                KEY:


     OPTIONS                            TRANSACTION MODES
A = PROJECT MESSAGE SENDING               1 - DELETE
B = PROJECT MESSAGE DISPLAY               2 - INITIATE
C = SESSION TERMINATION                   3 - REMOVE
D = TRANSACTION SELECTION                 4 - ADD
F = PROJECT / GROUP SWITCH                5 - UPDATE
H = USER MESSAGE SENDING                  6 - RETRIEVE
I = USER MESSAGE DISPLAY


                                       FOR OPTION - IDENTIFIER IS
                                            D     - TRANSACTION ID
                                            F     - PROJECT/GROUP
                                        A,B,C,H,I - (NOT USED)
```

Figure 48. Primary Option Menu (Display)

ENTER: OPTION: D (Transaction Selection)
TRANSACTION MODE: 5 (Update)

```
                        P R I M A R Y   M E N U

OPTION: D       TRANSACTION MODE: 5     IDENTIFIER:
                KEY:


     OPTIONS                            TRANSACTION MODES
A = PROJECT MESSAGE SENDING               1 - DELETE
B = PROJECT MESSAGE DISPLAY               2 - INITIATE
C = SESSION TERMINATION                   3 - REMOVE
D = TRANSACTION SELECTION                 4 - ADD
F = PROJECT / GROUP SWITCH                5 - UPDATE
H = USER MESSAGE SENDING                  6 - RETRIEVE
I = USER MESSAGE DISPLAY


                                       FOR OPTION - IDENTIFIER IS
                                            D     - TRANSACTION ID
                                            F     - PROJECT/GROUP
                                        A,B,C,H,I - (NOT USED)
```

Figure 49. Primary Option Menu (As Entered)

```
    S E C O N D A R Y   O P T I O N   S E L E C T I O N   PAGE:    1
ACTION:      (C=RETURN TO PRIMARY MENU; Q=EXIT TO SIGNON)       LAST
UPDATE     MODE:   SELECT:
KEY:


PA - PART SEGMENT
PD - STANDARD INFORMATION
IV - INVENTORY
CY - CYCLE COUNT
CD - CLOSE/DISBURSE INVENTORY
```

Figure 50. Secondary Option Menu (Display)

```
      ENTER:  SELECT: PD
```

```
    S E C O N D A R Y   O P T I O N   S E L E C T I O N   PAGE:    1
ACTION:      (C=RETURN TO PRIMARY MENU; Q=EXIT TO SIGNON)       LAST
UPDATE     MODE:   SELECT: PD
KEY:


PA - PART SEGMENT
PD - STANDARD INFORMATION
IV - INVENTORY
CY - CYCLE COUNT
CD - CLOSE/DISBURSE INVENTORY
```

Figure 51. Secondary Option Menu (As Entered)

The following screen specifies the transaction to access the Part Master and Standard Data segments.

```
                  S A M P L E   P R O B L E M
         P R I M A R Y   K E Y   S E L E C T I O N   S C R E E N
UPDATE                      TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 5PD   KEY:
                     ** ENTER THE FOLLOWING KEY INFORMATION **
          PART NUMBER-
          KEY FIELD---
```

Figure 52. Primary Key Selection (Display)

```
ENTER:  PART NUMBER- 02AN960C10
```

```
                  S A M P L E   P R O B L E M
         P R I M A R Y   K E Y   S E L E C T I O N   S C R E E N
UPDATE                      TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 5PD   KEY:
                     ** ENTER THE FOLLOWING KEY INFORMATION **
          PART NUMBER- 02AN960C10
          KEY FIELD---
```

Figure 53. Primary Key Selection (As Entered)

Since there is only one Standard Data segment for each Part Master, secondary key selection is not needed. The next screen displayed is the Data Display screen.

```
                  S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 5PD   KEY: 02AN960C10       02
              *** ENTER DATA FOR UPDATE ***
           PART NUMBER----- 02AN960C10
           DESCRIPTION----- WASHER
           PROC CODE------- 74
           INVENTORY CODE-- 2
           PLAN REV NO-----
           MAKE DEPT------- 1200
           COMM CODE------- 14
           RIGHT MAKE TIME-  6
           WRONG MAKE TIME-  6
```

Figure 54. Data Display for PD Transaction (Display)

```
      ENTER:  PLAN REV NO----- 88   (modify field)
              MAKE DEPT------- 1499 (modify field)
```

```
                  S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 5PD   KEY: 02AN960C10       02
              *** ENTER DATA FOR UPDATE ***
           PART NUMBER----- 02AN960C10
           DESCRIPTION----- WASHER
           PROC CODE------- 74
           INVENTORY CODE-- 2
           PLAN REV NO----- 88
           MAKE DEPT------- 1499
           COMM CODE------- 14
           RIGHT MAKE TIME-  6
           WRONG MAKE TIME-  6
```

Figure 55. Data Display for PD Transaction (As Entered)

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 5PD   KEY: 02AN960C10       02
            *** DATA MODIFIED SUCCESSFULLY ***
          PART NUMBER----- 02AN960C10
          DESCRIPTION----- WASHER
          PROC CODE------- 74
          INVENTORY CODE-- 2
          PLAN REV NO----- 88
          MAKE DEPT------- 1499
          COMM CODE------- 14
          RIGHT MAKE TIME-  6
          WRONG MAKE TIME-  6
```

Figure 56. Data Display for PD Transaction (Display)

The preceding screen shows that the updated segment was successfully modified in the database. Subsequent retrieval shows the segment with the fields modified.

```
ENTER:  TRX: 6CY
        KEY: 02RC07GF273J
```

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 6CY   KEY: 02RC07GF273J
            *** DATA MODIFIED SUCCESSFULLY ***
          PART NUMBER----- 02AN960C10
          DESCRIPTION----- WASHER
          PROC CODE------- 74
          INVENTORY CODE-- 2
          PLAN REV NO----- 88
          MAKE DEPT------- 1499
          COMM CODE------- 14
          RIGHT MAKE TIME-  6
          WRONG MAKE TIME-  6
```

Figure 57. Data Display Screen for PD Transaction (As Entered)

On this screen, the user is switching to transaction CY, transaction mode 6 (Retrieve), and Part Master (key 02RC07FG273J) without returning to the Primary or Secondary Option Menu.

```
        S E C O N D A R Y   K E Y   S E L E C T I O N

RETRIEVE       TRANSACTION: CYCLE COUNT
OPTION:       TRX: 6CY   KEY: 02RC07GF273J
SELECTION:         ** ENTER A SELECTION NUMBER FROM THIS SCREEN **
  INVENTORY    UNIT      CURRENT   ON     TOTAL      DISBURSEMENTS
  LOCATION     PRICE     REQMNTS  ORDER   STOCK   PLANNED  UNPLANNED
1 00 AK24527    2.40       33       0       33        1      700
2 0028009126     .00       17       0       17       57      700
3 0028011126     .00       26       0       26      240      729
```

Figure 58. Secondary Key Selection (Display)

Since there are multiple Stock Status segments for the specified Part Master, the choices are presented on a Secondary Key Selection screen.

```
ENTER:  SELECTION: 2
```

```
        S E C O N D A R Y   K E Y   S E L E C T I O N

RETRIEVE       TRANSACTION: CYCLE COUNT
OPTION:       TRX: 6CY   KEY: 02RC07GF273J
SELECTION: 2       ** ENTER A SELECTION NUMBER FROM THIS SCREEN **
  INVENTORY    UNIT      CURRENT   ON     TOTAL      DISBURSEMENTS
  LOCATION     PRICE     REQMNTS  ORDER   STOCK   PLANNED  UNPLANNED
1 00 AK24527    2.40       33       0       33        1      700
2 0028009126     .00       17       0       17       57      700
3 0028011126     .00       26       0       26      240      729
```

Figure 59. Secondary Key Selection (As Entered)

The second selection in the list is chosen.

```
                S A M P L E   P R O B L E M

RETRIEVE                    TRANSACTION: CYCLE COUNT
OPTION:       TRX: 6CY   KEY: 02RC07GF273J     0028009126    20

PART NUMBER---- 02RC07GF273J             DESCRIPTION---- RESISTOR
AREA----------- 2                        INV DEPT------- 80
PROJECT-------- 091                      DIVISION------- 26
UNIT PRICE-----          .00             UNIT----------- 0000
ATTR COAP------     0                    ATTR PLANNED---   0
ATTR COAD------ 0                        STOCK DATE----- 516
LAST TRANS----- 517                      RQMNTS CURRENT-       17
RQMNTS UNPLAN--         0                ON ORDER-------        0
TOTAL STOCK----        17                DISB PLAN------       57
DISB UNPLAN----       700                DISB SPARES----        0
DISB DIVERS----         0                PHYS COUNT-----       19
BOOK COUNT-----         0
```

Figure 60. Data Display for CY Transaction (Display)

Since mode 6 (Retrieve) was specified, the data on the preceding screen cannot be updated.

```
      ENTER:  TRX: 2PA
              KEY: 02XYZ
```

```
                S A M P L E   P R O B L E M

RETRIEVE                    TRANSACTION: CYCLE COUNT
OPTION:       TRX: 2PA   KEY: 02XYZ

PART NUMBER---- 02RC07GF273J             DESCRIPTION---- RESISTOR
AREA----------- 2                        INV DEPT------- 80
PROJECT-------- 091                      DIVISION------- 26
UNIT PRICE-----          .00             UNIT----------- 0000
ATTR COAP------     0                    ATTR PLANNED---   0
ATTR COAD------ 0                        STOCK DATE----- 516
LAST TRANS----- 517                      RQMNTS CURRENT-       17
RQMNTS UNPLAN--         0                ON ORDER-------        0
TOTAL STOCK----        17                DISB PLAN------       57
DISB UNPLAN----       700                DISB SPARES----        0
DISB DIVERS----         0                PHYS COUNT-----       19
BOOK COUNT-----         0
```

Figure 61. Data Display for CY Transaction (As Entered)

A new Part Master is to be added to the database.

```
                S A M P L E   P R O B L E M

INITIATE                    TRANSACTION: PART SEGMENT
OPTION:       TRX: 2PA   KEY: 02XYZ
            *** ENTER DATA FOR ADD ***
          PART NUMBER- 02XYZ
          DESCRIPTION- _____________________
```

Figure 62. Data Display for PA Transaction (Display)

This screen allows a description to be entered in the new Part Master segment.

```
      ENTER:  DESCRIPTION- CABLE
```

```
                S A M P L E   P R O B L E M

INITIATE                    TRANSACTION: PART SEGMENT
OPTION:       TRX: 2PA   KEY: 02XYZ
            *** ENTER DATA FOR ADD ***
          PART NUMBER- 02XYZ
          DESCRIPTION- CABLE_______________
```

Figure 63. Data Display for PA Transaction (As Entered)

```
                S A M P L E   P R O B L E M

INITIATE                    TRANSACTION: PART SEGMENT
OPTION:       TRX: 5PA   KEY: 02XYZ
            *** DATA ADDED SUCCESSFULLY ***
          PART NUMBER- 02XYZ
          DESCRIPTION- CABLE
```

Figure 64. Data Display for PA Transaction (Display)

The preceding screen specifies that the Part Master segment was added to the database.

```
ENTER:  TRX: 4PD
        KEY: blank out
```

```
                S A M P L E   P R O B L E M

INITIATE                    TRANSACTION: PART SEGMENT
OPTION:       TRX: 4PD   KEY:
            *** DATA ADDED SUCCESSFULLY ***
          PART NUMBER- 02XYZ
          DESCRIPTION- CABLE
```

Figure 65. Data Display for PA Transaction (As Entered)

A new Standard Data segment is to be added for the Part Master just added.

```
                S A M P L E   P R O B L E M
        P R I M A R Y   K E Y   S E L E C T I O N   S C R E E N
ADD                        TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 4PD   KEY:
                       ** ENTER THE FOLLOWING KEY INFORMATION **
          PART NUMBER-
          KEY FIELD---
```

Figure 66. Primary Key Selection (Display)

```
      ENTER:  PART NUMBER- 02XYZ
              KEY FIELD--- 02
```

```
                S A M P L E   P R O B L E M
        P R I M A R Y   K E Y   S E L E C T I O N   S C R E E N
ADD                        TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 4PD   KEY:
                       ** ENTER THE FOLLOWING KEY INFORMATION **
          PART NUMBER- 02XYZ
          KEY FIELD--- 02
```

Figure 67. Primary Key Selection (As Entered)

The preceding entry represents the full key for the Standard Data segment to be added.

```
                S A M P L E   P R O B L E M

ADD                          TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 4PD   KEY: 02XYZ            02
            *** ENTER DATA FOR ADD ***
          PART NUMBER----- 02XYZ
          DESCRIPTION----- CABLE
          PROC CODE------- __
          INVENTORY CODE-- _
          PLAN REV NO----- __
          MAKE DEPT------- ____
          COMM CODE------- ____
          RIGHT MAKE TIME-  0
          WRONG MAKE TIME-  0
```

Figure 68. Data Display for PD Transaction (Display)

```
ENTER:  PROC CODE------- 74
        INVENTORY CODE--  2
        PLAN REV NO-----  3
        MAKE DEPT------- 1200
        COMM CODE------- 72
        RIGHT MAKE TIME- 06
        WRONG MAKE TIME- 06
```

```
                S A M P L E   P R O B L E M

ADD                          TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 4PD   KEY: 02XYZ            02
            *** ENTER DATA FOR ADD ***
          PART NUMBER----- 02XYZ
          DESCRIPTION----- CABLE
          PROC CODE------- 74
          INVENTORY CODE-- 2
          PLAN REV NO----- 3_
          MAKE DEPT------- 1200
          COMM CODE------- 72__
          RIGHT MAKE TIME- 06
          WRONG MAKE TIME- 06
```

Figure 69. Data Display for PD Transaction (As Entered)

```
                  S A M P L E   P R O B L E M

ADD                          TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 4PD   KEY: 02XYZ            02
              *** DATA ADDED SUCCESSFULLY ***
           PART NUMBER----- 02XYZ
           DESCRIPTION----- CABLE
           PROC CODE------- 74
           INVENTORY CODE-- 2
           PLAN REV NO----- 3
           MAKE DEPT------- 1200
           COMM CODE------- 72
           RIGHT MAKE TIME-  6
           WRONG MAKE TIME-  6
```

Figure 70. Data Display for PD Transaction (Display)

The preceding screen specifies that the Standard Data segment was added to the database.

```
      ENTER:  TRX: 1PA
```

```
                  S A M P L E   P R O B L E M

ADD                          TRANSACTION: STANDARD INFORMATION
OPTION:       TRX: 1PA   KEY: 02XYZ            02
              *** DATA ADDED SUCCESSFULLY ***
           PART NUMBER----- 02XYZ
           DESCRIPTION----- CABLE
           PROC CODE------- 74
           INVENTORY CODE-- 2
           PLAN REV NO----- 3
           MAKE DEPT------- 1200
           COMM CODE------- 72
           RIGHT MAKE TIME-  6
           WRONG MAKE TIME-  6
```

Figure 71. Data Display for PD Transaction (As Entered)

The Part Master and Standard Data segments that were just added to the database are to be deleted.

A PA Data Display screen with the message `PRESS ENTER TO DELETE DATA` appears.

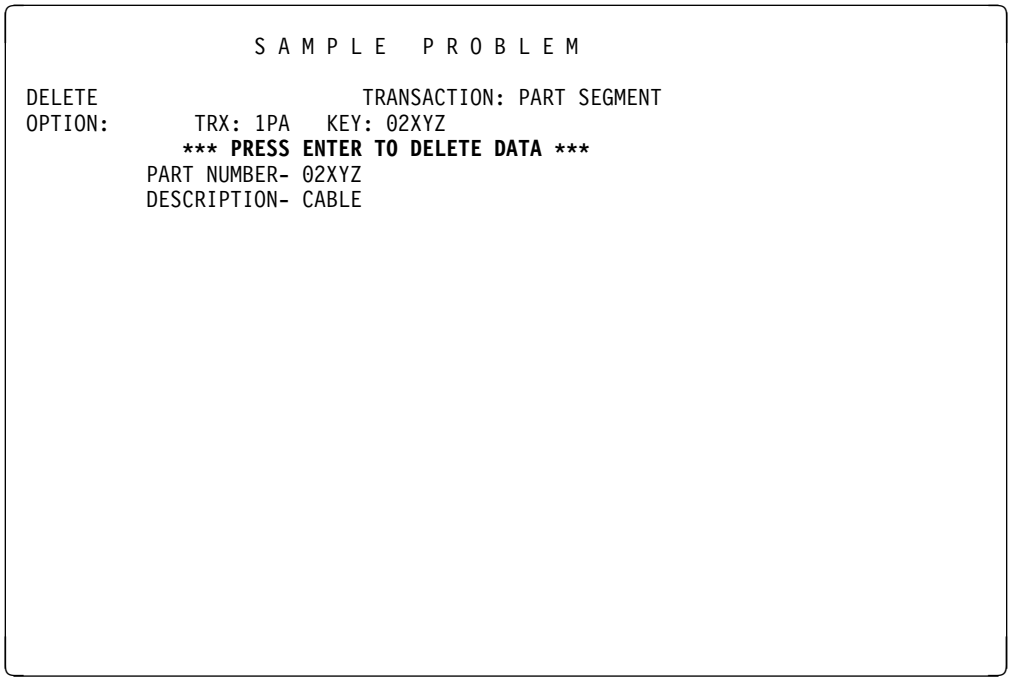

Figure 72. Data Display for PA Transaction (Display)

ENTER: Press the ENTER key

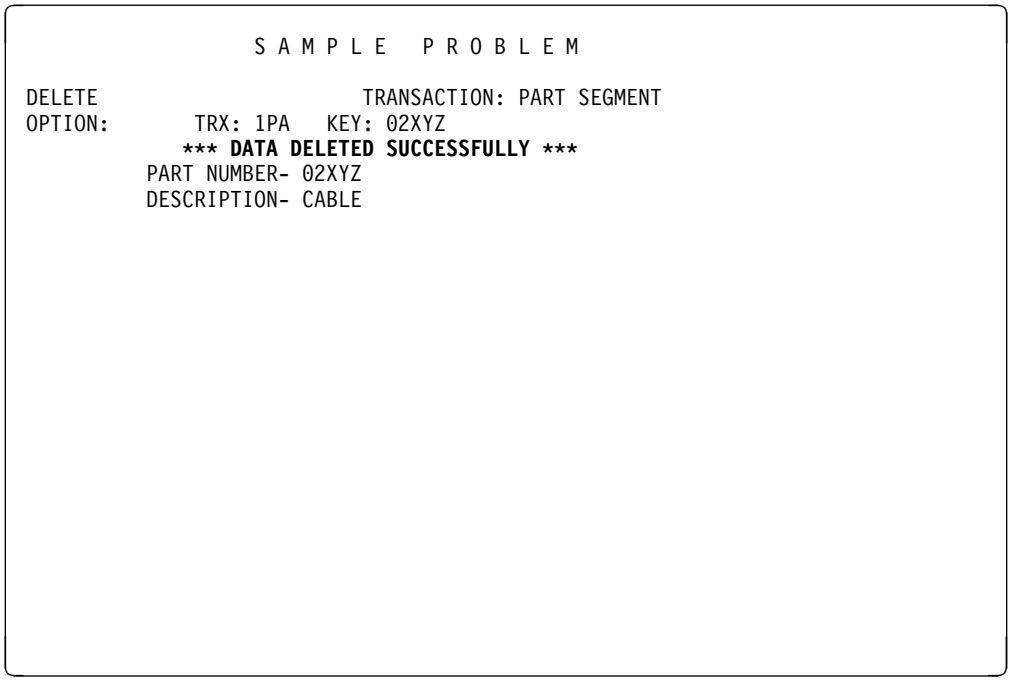

Figure 73. Data Display for PA Transaction (Display)

The preceding screen shows that Part Master 02XYZ was deleted from the database. In this case, the Standard Data segment that was added under the Part Master was also deleted.

```
ENTER:  TRX: 5CD
        KEY: blank out
```

```
                S A M P L E   P R O B L E M

DELETE                      TRANSACTION: PART SEGMENT
OPTION:       TRX: 5CD   KEY:
           *** DATA DELETED SUCCESSFULLY ***
          PART NUMBER- 02XYZ
          DESCRIPTION- CABLE
```

Figure 74. Data Display for PA Transaction (As Entered)

This screen selects the special processing transaction CD.

You also could have entered `OPTION: C` to return to the Primary Option Menu; then `OPTION: D`, `TRANSACTION MODE: 5`, and special processing `IDENTIFIER: CD` could be entered. The next screen displayed would be the Primary Key Selection screen for transaction CD.

```
                  S A M P L E   P R O B L E M
          P R I M A R Y   K E Y   S E L E C T I O N   S C R E E N
UPDATE                       TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION:       TRX: 5CD   KEY:
                        ** ENTER THE FOLLOWING KEY INFORMATION **
           PART NUMBER-
           00----------
           AREA--------
           INV DEPT----
           PROJECT-----
           DIVISION----
           FILLER------
```

Figure 75. Primary Key Selection (Display)

Part Number is the key of the Part Master segment. The remaining six fields combined produce Inventory Location, which is the key of the Stock Status segment.

```
ENTER:  PART NUMBER- 02AN960C10
```

```
                  S A M P L E   P R O B L E M
          P R I M A R Y   K E Y   S E L E C T I O N   S C R E E N
UPDATE                       TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION:       TRX: 5CD   KEY:
                        ** ENTER THE FOLLOWING KEY INFORMATION **
           PART NUMBER- 02AN960C10
           00----------
           AREA--------
           INV DEPT----
           PROJECT-----
           DIVISION----
           FILLER------
```

Figure 76. Primary Key Selection (As Entered)

```
        S E C O N D A R Y   K E Y   S E L E C T I O N

RETRIEVE       TRANSACTION: CYCLE COUNT
OPTION:       TRX: 5CD   KEY: 02AN960C10
SELECTION:         ** ENTER A SELECTION NUMBER FROM THIS SCREEN **
  INVENTORY    UNIT      CURRENT   ON     TOTAL      DISBURSEMENTS
  LOCATION     PRICE     REQMNTS  ORDER   STOCK   PLANNED  UNPLANNED
1 00 AA16511     .00      131      20      126      104        0
2 00 AK2877F     .00       88     -10       73       57        0
3 0028009126     .00      630       0      680     1053      104
```

Figure 77. Secondary Key Selection (Display)

The preceding screen shows all the Stock Status segments available under Part Master 02AN960C10. A properly designed Secondary Key Selection screen can often be used to display all information needed for an inquiry. This example shows a detailed breakdown by inventory location of the unit price, current requirements, stock on order, total stock on hand, and disbursements.

```
      ENTER:  SELECTION: 2
```

```
        S E C O N D A R Y   K E Y   S E L E C T I O N

RETRIEVE       TRANSACTION: CYCLE COUNT
OPTION:       TRX: 5CD   KEY: 02AN960C10
SELECTION: 2       ** ENTER A SELECTION NUMBER FROM THIS SCREEN **
  INVENTORY    UNIT      CURRENT   ON     TOTAL      DISBURSEMENTS
  LOCATION     PRICE     REQMNTS  ORDER   STOCK   PLANNED  UNPLANNED
1 00 AA16511     .00      131      20      126      104        0
2 00 AK2877F     .00       88     -10       73       57        0
3 0028009126     .00      630       0      680     1053      104
```

Figure 78. Secondary Key Selection (As Entered)

Inventory Location 00AK2877F is selected.

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION:       TRX: 5CD   KEY: 02AN960C10       00 AK2877F
     ENTER CLOSE|STOCK INCR|DISBURSE QUANTITIES-P|U
CLOSE ORDER-----         0                STOCK INCR------       0
DISBURSE QTY----         0                PLANNED/UNPLAN--
PART NUMBER----- 02AN960C10               DESCRIPTION----- WASHER
AREA------------                          INV DEPT-------- AK
PROJECT--------- 287                      DIVISION-------- 7F
UNIT PRICE------         .00              UNIT------------ EACH
ATTR COAP------- 270                      ATTR PLANNED----   0
ATTR COAD-------                          STOCK DATE------ ___
LAST TRANS------ 360                      RQMNTS CURRENT--      88
RQMNTS UNPLAN---         0                ON ORDER--------     -10
TOTAL STOCK-----        73                DISB PLAN-------      57
DISB UNPLAN-----         0                DISB SPARES-----       0
DISB DIVERS-----         0                PROC CODE------- 74
INVENTORY CODE-- 2                        PLAN REV NO----- 88
MAKE DEPT------- 1400                     COMM CODE------- 14
RIGHT MAKE TIME-    6                     WRONG MAKE TIME-  6
```

Figure 79. Data Display for CD Transaction (Display)

This screen is designed to accept input in the following fields:

- CLOSE ORDER
- STOCK INCR
- DISBURSE QTY
- PLANNED/UNPLANNED

The first three fields accept numeric quantities, while the last field accepts either P or U to designate planned or unplanned disbursements. If a character other than P or U is entered, an error message is displayed.

```
ENTER:  CLOSE ORDER-------- 0000010
        STOCK INCR--------- 0000005
        DISBURSE QTY------- 0000020
        PLANNED/UNPLANNED-- X
```

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION:       TRX: 5CD   KEY: 02AN960C10       00 AK2877F
     ENTER CLOSE|STOCK INCR|DISBURSE QUANTITIES-P|U
CLOSE ORDER----- 0000010                 STOCK INCR------ 0000005
DISBURSE QTY---- 0000020                 PLANNED/UNPLAN-- X
PART NUMBER----- 02AN960C10              DESCRIPTION----- WASHER
AREA------------ _                       INV DEPT-------- AK
PROJECT--------- 287                     DIVISION-------- 7F
UNIT PRICE------         .00             UNIT------------ EACH
ATTR COAP------- 270                     ATTR PLANNED----   0
ATTR COAD------- _                       STOCK DATE------ ___
LAST TRANS------ 360                     RQMNTS CURRENT--      88
RQMNTS UNPLAN---         0               ON ORDER--------     -10
TOTAL STOCK-----        73               DISB PLAN-------      57
DISB UNPLAN-----         0               DISB SPARES-----       0
DISB DIVERS-----         0               PROC CODE------- 74
INVENTORY CODE-- 2                       PLAN REV NO----- 88
MAKE DEPT------- 1400                    COMM CODE------- 14
RIGHT MAKE TIME-    6                    WRONG MAKE TIME-  6
```

Figure 80. Data Display for CD Transaction (As Entered)

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION:       TRX: 5CD   KEY: 02AN960C10       00 AK2877F
      9999 DISBURSEMENT CODE INCORRECT (X) SPECIFY P OR U
CLOSE ORDER-----        10               STOCK INCR------       5
DISBURSE QTY----        20               PLANNED/UNPLAN-- X
PART NUMBER----- 02AN960C10              DESCRIPTION----- WASHER
AREA------------ _                       INV DEPT-------- AK
PROJECT--------- 287                     DIVISION-------- 7F
UNIT PRICE------         .00             UNIT------------ EACH
ATTR COAP------- 270                     ATTR PLANNED----   0
ATTR COAD------- _                       STOCK DATE------ ___
LAST TRANS------ 360                     RQMNTS CURRENT--      88
RQMNTS UNPLAN---         0               ON ORDER--------     -10
TOTAL STOCK-----        73               DISB PLAN-------      57
DISB UNPLAN-----         0               DISB SPARES-----       0
DISB DIVERS-----         0               PROC CODE------- 74
INVENTORY CODE-- 2                       PLAN REV NO----- 88
MAKE DEPT------- 1400                    COMM CODE------- 14
RIGHT MAKE TIME-    6                    WRONG MAKE TIME-  6
```

Figure 81. Data Display for CD Transaction (error)

The preceding screen specifies that an error occurred on the prior input. In this case, an `X` was entered as the disbursement code instead of a `P` or a `U`.

If multiple errors were detected after input, the message `ENTER "E" TO DISPLAY ERROR MESSAGES` would have appeared instead.

ENTER: OPTION: E

```
                S A M P L E   P R O B L E M

UPDATE                       TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION: E     TRX: 5CD   KEY: 02AN960C10       00 AK2877F
       9999 DISBURSEMENT CODE INCORRECT (X) SPECIFY P OR U
CLOSE ORDER-----        10                STOCK INCR------       5
DISBURSE QTY----        20                PLANNED/UNPLAN-- X
PART NUMBER----- 02AN960C10               DESCRIPTION----- WASHER
AREA------------ _                        INV DEPT-------- AK
PROJECT--------- 287                      DIVISION-------- 7F
UNIT PRICE------         .00              UNIT------------ EACH
ATTR COAP------- 270                      ATTR PLANNED----   0
ATTR COAD------- _                        STOCK DATE------ ___
LAST TRANS------ 360                      RQMNTS CURRENT--      88
RQMNTS UNPLAN---         0                ON ORDER--------     -10
TOTAL STOCK-----        73                DISB PLAN-------      57
DISB UNPLAN-----         0                DISB SPARES-----       0
DISB DIVERS-----         0                PROC CODE------- 74
INVENTORY CODE-- 2                        PLAN REV NO----- 88
MAKE DEPT------- 1400                     COMM CODE------- 14
RIGHT MAKE TIME-    6                     WRONG MAKE TIME-  6
```

Figure 82. Data Display Screen for CD Transaction (As Entered)

```
                  E R R O R   M E S S A G E S

OPTION:                                                  PAGE: 001
                                                              LAST

9999 DISBURSEMENT CODE INCORRECT (X) SPECIFY P OR U
```

Figure 83. Error Message Screen (Display)

The preceding error message describes the error that was detected.

ENTER: Press the ENTER key

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION:       TRX: 5CD   KEY: 02AN960C10       00 AK2877F
       9999 DISBURSEMENT CODE INCORRECT (X) SPECIFY P OR U
CLOSE ORDER-----        10                STOCK INCR------       5
DISBURSE QTY----        20                PLANNED/UNPLAN-- X
PART NUMBER----- 02AN960C10               DESCRIPTION----- WASHER
AREA------------ _                        INV DEPT-------- AK
PROJECT--------- 287                      DIVISION-------- 7F
UNIT PRICE------         .00              UNIT------------ EACH
ATTR COAP------- 270                      ATTR PLANNED----   0
ATTR COAD------- _                        STOCK DATE------ ___
LAST TRANS------ 360                      RQMNTS CURRENT--      88
RQMNTS UNPLAN---         0                ON ORDER--------     -10
TOTAL STOCK-----        73                DISB PLAN-------      57
DISB UNPLAN-----         0                DISB SPARES-----       0
DISB DIVERS-----         0                PROC CODE------- 74
INVENTORY CODE-- 2                        PLAN REV NO----- 88
MAKE DEPT------- 1400                     COMM CODE------- 14
RIGHT MAKE TIME-    6                     WRONG MAKE TIME-  6
```

Figure 84. Data Display Screen for CD Transaction (Display)

The preceding screen is redisplayed to let you correct the error.

```
      ENTER:  PLANNED/UNPLANNED-- P
```

This will correct the Planned/Unplanned field (Disbursement Code).

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION:       TRX: 5CD   KEY: 02AN960C10       00 AK2877F
       9999 DISBURSEMENT CODE INCORRECT (X) SPECIFY P OR U
CLOSE ORDER-----        10                STOCK INCR------       5
DISBURSE QTY----        20                PLANNED/UNPLAN-- P
PART NUMBER----- 02AN960C10               DESCRIPTION----- WASHER
AREA------------ _                        INV DEPT-------- AK
PROJECT--------- 287                      DIVISION-------- 7F
UNIT PRICE------         .00              UNIT------------ EACH
ATTR COAP------- 270                      ATTR PLANNED----   0
ATTR COAD------- _                        STOCK DATE------ ___
LAST TRANS------ 360                      RQMNTS CURRENT--      88
RQMNTS UNPLAN---         0                ON ORDER--------     -10
TOTAL STOCK-----        73                DISB PLAN-------      57
DISB UNPLAN-----         0                DISB SPARES-----       0
DISB DIVERS-----         0                PROC CODE------- 74
INVENTORY CODE-- 2                        PLAN REV NO----- 88
MAKE DEPT------- 1400                     COMM CODE------- 14
RIGHT MAKE TIME-    6                     WRONG MAKE TIME-  6
```

Figure 85. Data Display for CD Transaction (As Entered)

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION:       TRX: 5CD   KEY: 02AN960C10       00 AK2877F
     *** WARNING: NEGATIVE ON-ORDER POSITION ***
CLOSE ORDER-----        10                STOCK INCR------       5
DISBURSE QTY----        20                PLANNED/UNPLAN-- P
PART NUMBER----- 02AN960C10               DESCRIPTION----- WASHER
AREA------------ _                        INV DEPT-------- AK
PROJECT--------- 287                      DIVISION-------- 7F
UNIT PRICE------         .00              UNIT------------ EACH
ATTR COAP------- 270                      ATTR PLANNED----   0
ATTR COAD------- _                        STOCK DATE------ ___
LAST TRANS------ 360                      RQMNTS CURRENT--      88
RQMNTS UNPLAN---         0                ON ORDER--------     -20
TOTAL STOCK-----        58                DISB PLAN-------      77
DISB UNPLAN-----         0                DISB SPARES-----       0
DISB DIVERS-----         0                PROC CODE------- 74
INVENTORY CODE-- 2                        PLAN REV NO----- 88
MAKE DEPT------- 1400                     COMM CODE------- 14
RIGHT MAKE TIME-    6                     WRONG MAKE TIME-  6
```

Figure 86. Data Display for CD Transaction (Display)

The preceding screen signifies that the entered data was validated and the segment was updated in the database. The `ON ORDER`, `DISP`, and `TOTAL STOCK` fields reflect the new inventory status in the segment.

The warning message displayed on this screen notifies you that a negative on-order position exists (`ON ORDER--------     -20`).

This completes the sample problem.

You may want to try additional operations with the transactions supplied to become more familiar with the system. Keep in mind that additions, deletions, and updates modify the database. As a result, a subsequent run of the sample problem may not produce the same results.

To terminate the sample problem:

```
ENTER:  OPTION: Q
```

```
                S A M P L E   P R O B L E M

UPDATE                      TRANSACTION: CLOSE/DISBURSE INVENTORY
OPTION: Q     TRX: 5CD   KEY: 02AN960C10       00 AK2877F
     *** WARNING: NEGATIVE ON-ORDER POSITION ***
CLOSE ORDER-----        10               STOCK INCR------       5
DISBURSE QTY----        20               PLANNED/UNPLAN-- P
PART NUMBER----- 02AN960C10              DESCRIPTION----- WASHER
AREA------------ _                       INV DEPT-------- AK
PROJECT--------- 287                     DIVISION-------- 7F
UNIT PRICE------         .00             UNIT------------ EACH
ATTR COAP------- 270                     ATTR PLANNED----   0
ATTR COAD------- _                       STOCK DATE------ ___
LAST TRANS------ 360                     RQMNTS CURRENT--      88
RQMNTS UNPLAN---         0               ON ORDER--------     -20
TOTAL STOCK-----        58               DISB PLAN-------      77
DISB UNPLAN-----         0               DISB SPARES-----       0
DISB DIVERS-----         0               PROC CODE------- 74
INVENTORY CODE-- 2                       PLAN REV NO----- 88
MAKE DEPT------- 1400                    COMM CODE------- 14
RIGHT MAKE TIME-    6                    WRONG MAKE TIME-  6
```

Figure 87. Terminating the Sample Problem

This terminates the session.

Chapter 12. Installation Macro Messages

The messages that originate in the installation process are listed in this chapter. They come from the following macros:

- DEFADF (invoked by jobs ADFOPT and ADFUSER)
- ADFMFS (invoked by job SCREENS)

Most of the messages are self-explanatory. To correct errors, see the description of the designated parameter in Chapter 3, “Customization Macros (DEFADF and ADFMFS).”

Some messages may contain a word prefixed by an ampersand (&), as in &USRLANG. With these messages, you can expect to see a value; the value is normally that of a parameter you coded in the macro invocation and that is named in the affected message.

Additional messages for the various components of IMSADF II are listed in the related publications: *IMS Application Development Facility II Version 2 Release 2 User Reference* and *IMS Application Development Facility II Version 2 Release 2 Application Development Reference*.

ADFM601 ADFID NOT SPECIFIED -- "MFC1" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM602 ADFID MUST BE 4 CHARS WITH NO QUOTES

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM603 FIRST CHAR OF ADFID MUST BE NON-NUMERIC

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM604 MFSTRLR NOT SPECIFIED -- "1" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM605 MFSTRLR LIMITED TO 1 CHARACTER

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM606 TRXTRLR LIMITED TO 1 CHARACTER

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM607 TRXTRLR VALUE OF ZERO IS NOT ALLOWED

Explanation: Zero is reserved for non-conversational transaction trailer use.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM608 WRKSEGS NOT SPECIFIED -- "0" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM609 NON-NUMERIC VALUE IN WRKSEGS

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM610 WRKSEGS MAX VALUE IS 15

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM611 WRKSIZE NOT SPECIFIED -- "6000" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM612 NON-NUMERIC VALUE IN WRKSIZE

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM613 WRKAREA ENTERED TOO LARGE--USING MAX &MAXSPA

Explanation: Your WRKAREA was larger than the maximum size shown. The maximum value is used.

System Action: This is a warning message; generation continues.

ADFM614 SPA SIZE TOO LARGE -- MAXIMUM IS &MAXSPA

Explanation: Your WRKAREA value is less than 2000 or less than WRKSIZE.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM615 SPA DB TOO SMALL -- MINIMUM IS &SPASIZE

Explanation: WRKSEGS times WRKSIZE is less than the mimimum size WRKAREA.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Increase your WRKAREA value. You may need to change WRKSIZE and/or WRKSEGS. Rerun the job.

ADFM616 WRKSIZE TIMES WRKSEGS GREATER THAN WRKAREA &MAXSPA

Explanation: WRKSIZE times WRKSEGS must be less than or equal to the WRKAREA value.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM617 DEVNAME NOT SPECIFIED -- "(2)" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM618 DEVTYPE NOT SPECIFIED -- "(2)" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM619 DEVTYPE INPUT &DEVTYPE(&DEVI) MUST BE 2-9

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM620 DEVCHRS NOT SPECIFIED-- "(0)" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM621 DEVCHRS INPUT &DEVCHRS(&DEVI) MUST RANGE 0-22

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM622 NCPRINT NOT SPECIFIED -- "PRINTER" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM623 NCPRINT LIMITED TO 8 CHARS WITH NO QUOTES

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM624 ADFNODE NOT SPECIFIED -- "IMSADF" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM625 ADFNODE LENGTH EXCEEDS &NODELEN CHARS

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM626 NEWADF NOT SPECIFIED -- "IMSADF" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM627 NEWADF LENGTH EXCEEDS &NODELEN CHARS

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM628 OLDADF NOT SPECIFIED -- "IMSADF" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM629 OLDADF LENGTH EXCEEDS &NODELEN CHARS

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM630 IMSNODE NOT SPECIFIED -- "IMSVS" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM631 ASMBLR NOT SPECIFIED -- "F" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM632 ASMBLR MUST BE "F" OR "H"

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM633 V50RES NOT SPECIFIED -- "YES" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM634 V50RES MUST BE "YES" OR "NO"

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM635 DATEFMT NOT SPECIFIED -- "U" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM636 DATEFMT MUST BE "S,B,U,E OR O"

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM637 KANAME NOT SPECIFIED -- "STD" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM638 KANAME MUST BE "ALT OR STD"

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM639 COMPMOD NOT SPECIFIED -- "NO" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM640 COMPMOD MUST BE "YES" "NO" OR "ALT"

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM641 RULESBL NOT SPECIFIED -- "0" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM642 NON-NUMERIC VALUE IN RULESBL

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM643 COMPMBL NOT SPECIFIED -- "0" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM644 NON-NUMERIC VALUE IN COMPMBL

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM645 RGLIB NOT SPECIFIED -- "L" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM646 RGLIB MUST BE "L OR O"

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM647 &USRLANG IS NOT A VALID PRIMARY LANGUAGE

Explanation: USRLANG must be a single character A-Z or 0-9, but not S or I.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM648 &ALTLANG(&J) IS NOT A VALID ALTERNATE LANGUAGE

Explanation: ALTLANG values must be a single character A-Z or 0-9, but not S or I. If you are generating multiple alternate languages, make sure your language and SYSIDs are properly delimited.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM649 SYSID (&ALTLANG(&J+1)) WITH LANGUAGE (&ALTLANG(&J)) MUST BE FOUR CHARACTERS

Explanation: If you are generating multiple alternate languages, make sure your language and SYSIDs are properly delimited.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM650 NON-NUMERIC VALUE IN COMMLEN

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM651 COMMLEN SIZE TOO LARGE-- MAX IS &MAXAVAL

Explanation: The maximum available value for COMMLEN is the WRKAREA value you specified minus the fixed portion of it, which is used by IMSADF II. The fixed portion is currently 800 bytes.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM652 RACF NOT SPECIFIED -- "TERMINAL" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM653 RACF MUST BE "RACFUSER" OR "TERMINAL" OR "RULEGEN"

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM654 TRMDFLT VALUE NOT CODED; MINIMUM 1 SET REQUIRED THE FIRST DEVNAME, DEVTYPE AND DEVCHRS NAME MAKE UP YOUR DEFAULT SET OF TERMINAL CHARACTERISTICS.

Explanation: Self-explanatory

System Action: This is a warning message; generation continues.

ADFM655 TRMDFLT MUST BE 1 THRU 5, INCLUSIVE

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM656 YOU MUST SPECIFY AT LEAST &DEVDFCT DEVCHRS VALUES

Explanation: The number of DEVCHRS parameters must be the same as the value of TRMDFLT.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM657 YOU MUST SPECIFY AT LEAST &DEVDFCT DEVNAME VALUES

Explanation: The number of DEVNAME parameters must be the same as the value of TRMDFLT.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM658 YOU MUST SPECIFY AT LEAST &DEVDFCT DEVTYPE VALUES

Explanation: The number of DEVTYPE parameters must be the same as the value of TRMDFLT.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM659 NO. OF DEVNAMES MUST BE SAME AS NO. OF DEVTYPES

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM660 NO. OF DEVNAMES MUST BE SAME AS NO. OF DEVCHRS

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM661 DELIM NOT SPECIFIED -- "$$" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM662 DELIM MUST BE 2 CHARS OR 4 WITH QUOTES

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM663 WRKREDU NOT SPECIFIED -- "NO" USED

Explanation: Self-explanatory

System Action: Default values are assumed.

ADFM664 WRKREDU CAN ONLY BE USED WITH WORK DATABASE

Explanation: You have selected a large IMS SPA by choosing `WRKSEGS=0`. You also selected `WRKREDU=YES`, but it is only available for the Work database option.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM665 WRKREDU MUST BE "YES","Y","NO" OR "N"

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM666 RULESVS NOT SPECIFIED -- "0" USED

Explanation: A BLDL and LOAD will be performed for each rule.

System Action: This is a warning message; generation continues.

ADFM667 NON-NUMERIC VALUE IN RULESVS; NOT E, K OR M

Explanation: Values entered may be numeric followed only by the above letters, representing dynamic rules storage in:

n--number of bytes
nE--scientific notation
nK--n thousand bytes
nM--n megabytes

You entered another letter.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM668 IMSTEST NOT SPECIFIED -- "IMSVS" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM669 SMPNODE NOT SPECIFIED -- "IMSVS" USED

Explanation: Self-explanatory. The SMP data sets default to those of IMS.

System Action: Default values are assumed; generation continues.

ADFM670 ENVIR NOT SPECIFIED -- "IMS" USED

Explanation: Self-explanatory

System Action: Default values are assumed; generation continues.

ADFM671 ENVIR MUST BE "IMS" OR FIRST CHARACTER. &ENVIR WAS CODED

Explanation: Self-explanatory

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM672 SPATS NOT SPECIFIED -- "AUX" USED

Explanation: AUXiliary is the default.

System Action: Default values are assumed; generation continues.

ADFM673 SPATS MUST BE "AUX" OR "MAIN", &SPATS WAS CODED

Explanation: Values other than AUX or MAIN were entered.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM678 DB2DB NOT SPECIFIED -- "NO" USED

Explanation: Self-explanatory.

System Action: Default values are assumed; generation continues.

ADFM679 DB2DB MUST BE "YES","Y","NO" OR "N"

Explanation: Other values were entered.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM681 DB2 WORK TABLE IS &SEGNUM SEGMENTS FOR A TOTAL OF &MAXSPA BYTES

Explanation: This is an informational message. No matter how many segments you enter, no more than two are used.

System Action: Default values are assumed; generation continues.

ADFM682 NON-NUMERIC VALUE IN MCTID

Explanation: MCTID must be numeric and less than 200

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM683 MCTID VALUE MUST BE LESS THAN 200

Explanation: MCTID must be numeric and less than 200

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM684 ONLY ''DBCS'' MAY BE ENTERED IN ADDITION TO THE PRIMARY LANGUAGE. (&TPRI)'

Explanation: Something other than the literal DBCS was entered.

System Action: DEFADF macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

ADFM701 ''DEVNAME'' AND ''DEVTYPE'' ARE INCOMPATIBLE

Explanation: The number of DEVNAME and DEVTYPE parameters are not the same.

System Action: ADFMFS macro generation is terminated.

Programmer Response: Correct the error, and rerun the job.

ADFM702 &C IS AN UNKNOWN DEVICE NAME

Explanation: The variable(s) shown is not in the list of DEVNAME values.

System Action: ADFMFS macro generation is terminated.

Programmer Response: Correct the error, and rerun the job.

ADFM703 &C IS AN UNKNOWN DEVICE TYPE

Explanation: The variable(s) shown is not in the list of DEVTYPE values.

System Action: ADFMFS macro generation is terminated.

Programmer Response: Correct the error, and rerun the job.

ADFM704 USRLANG=&USRLANG IS NOT A VALID LANGUAGE

Explanation: USRLANG must be a single character A-Z or 0-9, but not S or I.

System Action: ADFMFS macro generation is terminated after other parameters are validated.

Programmer Response: Correct the error, and rerun the job.

Appendix A. Sample IMSGEN Control Statements

The following control statements need to be added to your IMSGEN SYSIN stream to activate IMSADF II in your IMS environment.

```
//*********************************************************************
//*                                                                   *
//*  THIS SECTION PROVIDES THE NECESSARY IMS/ESA SYSTEM DEFINITION    *
//*  CONTROL STATEMENTS TO INSTALL IMSADF II IN AN EXISTING           *
//*  IMS/ESA SYSTEM.                                                  *
//*                                                                   *
//*  NOTE:                                                            *
//*                                                                   *
//*  THIS IS A SAMPLE ONLY.  REFER TO THE IMSADF II INSTALLATION GUIDE *
//*  AND THE IMS/ESA INSTALLATION GUIDE FOR SPECIFIC INFORMATION      *
//*                                                                   *
//*                                                                   *
//*********************************************************************

     DATABASE  DBD=MFDPAR01 *              AUDIT
     DATABASE  DBD=MFDPSP01 *              SIGN-ON PROFILE
     DATABASE  DBD=MFDPMS01 *              MESSAGE
     DATABASE  DBD=MFC1WORK                WORK DATA BASE (OPTIONAL)
     APPLCTN   PSB=MFC1TOM,PGMTYPE=TP      SIGN-ON/OPTION MENUS
     TRANSACT  CODE=MFC1T01,MODE=SNGL,SPA=(28,CORE,FIXED)
     TRANSACT  CODE=MFC1T02,MODE=SNGL,SPA=(28,CORE,FIXED)
     TRANSACT  CODE=MFC1T03,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=MFC1TCT,PGMTYPE=TP      COMMON CONV TRX DRIVER
     TRANSACT  CODE=MFC1TCT,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=MFC1T99,PGMTYPE=TP      TERMINATION
     TRANSACT  CODE=MFC1T99,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=MFC1V01,PGMTYPE=TP      PROJECT MESSAGE SENDING
     TRANSACT  CODE=MFC1V01,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=MFC1V05,PGMTYPE=TP      PROJECT MESSAGE DISPLAY
     TRANSACT  CODE=MFC1V05,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=MFC1V06,PGMTYPE=TP      USER MESSAGE SENDING
     TRANSACT  CODE=MFC1V06,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=MFC1V07,PGMTYPE=TP      USER MESSAGE DISPLAY
     TRANSACT  CODE=MFC1V07,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=SAMPTOR,PGMTYPE=TP      SAMPLE TRANSACTION DRIVER
     TRANSACT  CODE=SAMPTOR,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=SAMPVCD,PGMTYPE=TP     SAMPLE SPECIAL PROCESSING
     TRANSACT  CODE=SAMPVCD,MODE=SNGL,SPA=(28,CORE,FIXED)
     APPLCTN   PSB=MFC1BCTP,PGMTYPE=BATCH  DATA BASE UPDATE
     APPLCTN   PSB=MFC1BDDR,PGMTYPE=BATCH  RDOC

//***END OF IMSADF II INSTALLATION JCL*********************************
```

Note: If the installed ADFID is not MFC1, the DBD names of the DL/I IMSADF II databases are:

```
DBD  = ????AUDT        AUDIT
DBD  = ????SIGN        SIGN-ON PROFILE
DBD  = ????MSGS        MESSAGE
```

where ???? = ADFID

Replace MFC1 with the ADFID on all other specifications as well.

This example assumes you are using a SPA/HDAM Work database combination, rather than a SPA-only IMSADF II system. It also assumes the default of DL/I for the supplied IMSADF II databases.

Appendix B. Sample Problem Examples

This appendix contains the following:

- Sample problem JCL
- User-written audit exit for the sample problem
- Sample special processing routines (written in COBOL and PL/I, includes scratch pad area layouts)
- Layout of scratch pad area (written in Assembler)

Sample Problem JCL

The following statements must be added to the IMS control region JCL (assuming `ADFID=MFC1` and the environment is DL/I):

```
//MFDPAR01    DD  DSN=IMSADF.MFDPAR01.DB,DISP=OLD        AUDIT DB
//MFDPMS01    DD  DSN=IMSADF.MFDPMS01.DB,DISP=OLD        MESSAGE DB
//MFDPSP01    DD  DSN=IMSADF.MFDPSP01.DB,DISP=OLD        SIGN-ON DB
//MFC1WORK    DD  DSN=IMSADF.MFC1WORK.DB,DISP=OLD        SPA-WORK DB
```

DD statements for the IMS sample databases must also be present.

The following statements must be added to the STEPLIB concatenation in the JCL for the message region(s) in which IMSADF II applications will execute:

```
//             DD  DSN=IMSADF.RULLIB,DISP=SHR         RULES LIBRARY
//             DD  DSN=IMSADF.ADFLOAD,DISP=SHR       PROGRAM LIBRARY
```

User-Written Audit Exit for the Sample Problem

```
*********************************************************************
* THIS ROUTINE IS USED IN THE SAMPLE PROBLEM AS A USER AUDIT EXIT.     *
* THE ROUTINE CHECKS THE DISBURSEMENT CODE ENTERED FOR A 'P' OR        *
* A 'U'. IF VALID, THE RETURN IS TRUE.  IF NOT, THE RETURN IS FALSE.   *
* THIS ROUTINE IS CALLED BY THE AUDITOR WHEN AN AUDIT DESCRIPTOR       *
* CODE GREATER THAN 69 IS RECOGNIZED.                                  *
*********************************************************************
USERAUDT START 0
         STM   14,12,12(13)          SAVE REGISTERS
         BALR  12,0
         USING *,12
         ST    13,SAVE+4
         LA    10,SAVE
         ST    10,8(13)
         LR    13,10
         SPACE
*  CHECK DESCRIPTOR CODE FOR 75 -- IF NOT RETURN
         LR    3,1                   SAVE PARM REG POINTER
         L     9,20(3)
         NI    0(9),X'0              SET SW TO FALSE
         SPACE
         L     4,8(3)                POINTER TO PCODE
         CLC   =C'75',0(4)           DOES PCODE =75
         BE    OK1                   YES
         B     RETURN
OK1      L     5,36(3)               KEY FIELD OF OPERATION DESC.
         L     6.12(3)               AUDIT DATA BASE PCB
         CALL  SEGHNDLR,(DF,GN,(5),FE,(6),AREA),VL
         L     7,0(3)                POINTER TO FIELD
         L     8,4(3)                POINTER TO FIELD DESCRIPTION
         SR    9,9
         LH    9,9(8)                FIELD LENGTH
         BCTR  9,0                   DECREMENT BY 1
         STC   9,COMP+1              STORE IN COMPARE INSTRUCTION
         STC   9,COMP1+1
COMP     CLC   0(1,7),AREA+5         CHECK FOR FIRST VALUE
         BE    TRUE
COMP1    CLC   0(1,7),AREA+7         CHECK FOR SECOND VALUE
         BE    TRUE
         B     RETURN
TRUE     L     9,20(3)               POINTER TO TRUE BYTE
         OI    0(9),X'80'            SET ON TRUE
RETURN   L     13,SAVE+4
         LM    14,12,12(13)          RETURN REGISTERS
         BR    14
SAVE     DC    18F'0'
AREA     DC    10C'AREA'
DF       DC    C'DF'
GN       DC    C'GN  '
FE       DC    C'FE'
         END   USERAUDT
```

Sample Special Processing Routines

This section contains listings that support special processing routines. The listings are written in COBOL and PL/I. Both the COBOL and PL/I routines do the same thing: they support the close/disburse function of the IMS sample problem using the Special Processing Interface routine.

The listings are:

- COBOL source for SAMPUCD (the close/disburse routine)
- SPA layout for COBOL (SPACOBOL in IMSADF.ADFMAC)
- PL/I source for SAMPUCD (the SPA layout is included in this source CE listing and in the MACLIB as SPAPLI)
- The Rule Generator statements to create a load module incorporating SAMPUCD with the conversational driver base load module.

  Note: The SYSLIB data set for the Rule Generator should include either a COBOL or PL/I call library.

COBOL Source Listing for SAMPUCD

```
IDENTIFICATION DIVISION.
 PROGRAM-ID.
    COBOLPGM.
 DATE-COMPILED. JUNE 20, 1977.
 REMARKS.
    SAMPLE COBOL PROGRAM

ENVIRONMENT DIVISION
 CONFIGURATION SECTION.
  SOURCE-COMPUTER. IBM-370.
  OBJECT-COMPUTER. IBM-370.

DATA DIVISION.

WORKING-STORAGE SECTION.
77  OPT               PICTURE 9(9)     COMPUTATIONAL.
77  GOTPD             PICTURE X        VALUE 'N'.
77  ECODE             PICTURE 9(9)     COMPUTATIONAL VALUE 0.
77  AUDITINFO         PICTURE 9(9)     COMPUTATIONAL.
77  IV                PICTURE XX       VALUE 'IV'.
77  HREP              PICTURE X(4)     VALUE 'HREP'.
77  PD                PICTURE XX       VALUE 'PD'.
77  GU                PICTURE X(4)     VALUE 'GU  '.
77  PCBNO             PICTUREX.
77  COKEYLEN          PICTURE 9(5)  COMPUTATIONAL.
01  COKEY.
  03  FILLER          PICTURE X(17).
  03  COKEY-PD        PICTURE XX.
  03  FILLER          PICTURE X(31).
01KEYOP.
  03  TYPEKEY         PICTURE X.
    88  FULLKEY       VALUE 'F'.
    88  PARTKEY       VALUE 'P'.
  03  KEYREL          PICTURE X.
    88  EQ            VALUE 'E'.
    88  GT            VALUE 'G'.
EJECT
```

```
*****************************************************************
*                                                               *
*  PARAMETER LIST FOR THE SEGMENT HANDLER                       *
*                                                               *
*  REGISTER 1 POINTING TO A LIST CONSISTING OF THE              *
*  FOLLOWING ADDRESS ON ENTRY.                                  *
*                                                               *
*       1  ID OF SEGMENT DESIRED.                               *
*       2  DLI FUNCTION DESIRED.                                *
*       3  KEY DESIRED (OPTIONAL).  IF KEY IS NOT               *
*            SPECIFIED, THE DEFAULT IS THE LAST KEY READ.       *
*       4  KEY OPERATOR (OPTIONAL).  TWO CHARACTERS:            *
*            FIRST CHARACTER: F/P (FULL OR PARTIAL).            *
*              F IS THE DEFAULT.                                *
*            SECOND CHARACTER: E/G (EQUAL OR GREATER).          *
*                     E IS THE DEFAULT.                         *
*              5  PCB# (OPTIONAL).  USER PCB NUMBER (1-5).      *
*              6  I/O AREA ADDRESS (OPTIONAL).                  *
*                   DEFAULTS TO ADDRESS OF READ AREA IN SPA.    *
*                                                               *
*     NOTE: SPARTNCD IS THE RETURN CODE AREA IN THE SPA.        *
*           SPADLIST IS THE DL/I RETURN CODE AREA IN THE SPA.   *
*           THESE FIELDS ARE NOT PASSED IN THE CALL LIST, BUT   *
*           THEY SHOULD BE CHECKED AFTER RETURN FROM THE CALLED *
*           PROGRAM.                                            *

   SKIP3

*****************************************************************
*                                                               *
*  PARAMETER LIST TO INVOKE THE SERVICE ROUTINE DATA MAPPER     *
*                                                               *
*    REGISTER 1 POINTING TO A LIST CONSISTING OF THE            *
*     FOLLOWING ADDRESSES ON ENTRY:                             *
*                                                               *
*          1  2-BYTE CHARACTER STRING SIGNIFYING NAME OF        *
*             MAPPING RULE MODULE.                              *
*          2  MAPPING AREA IN PROGRAM ADDRESS, (WORKURCD)       *
*          3  MAPPING OPTION ADDRESS,                           *
*                           1 = MAP INTO SERVICE RTN            *
*                           2 = MAP OUT OF SERVICE RTN          *
*                                                               *
*      NOTE:    SPARTNCD IS THE RETURN CODE AREA IN THE SPA.    *
*               THIS FIELD IS NOT PASSED IN THE CALL LIST, BUT  *
*               IT SHOULD BE CHECKED UPON RETURN FROM THE CALLED*
*               PROGRAM.                                        *
*                                                               *

 SKIP3
```

```
*****************************************************************
*                                                               *
*  PARAMETER LIST TO INVOKE THE AUDITOR                         *
*                                                               *
*                                                               *
*  NOTE     SPARTNCD IS THE RETURN CODE AREA IN THE SPA.        *
*           THIS FIELD IS NOT PASSED IN THE CALL LIST, BUT      *
*           IT SHOULD BE CHECKED UPON RETURN FROM THE CALLED    *
*           PROGRAM.                                            *
*                                                               *
*****************************************************************
    EJECT
*    MAPPED INPUT FIELDS - DATA MAPPER WORK AREA
 01  WORKURCD.
    03  SACDCLOR     PICTURE S9(9) COMPUTATIONAL.
    03  SACDCLST     PICTURE S9(9) COMPUTATIONAL.
    03  SACDDQTY     PICTURE S9(9) COMPUTATIONAL.
    03  SAIVONOR     PICTURE S9(9) COMPUTATIONAL.
    03  SAIVSTCK     PICTURE S9(9) COMPUTATIONAL.
    03  SAIVDIPL     PICTURE S9(9) COMPUTATIONAL.
    03  SAIVDIUN     PICTURE S9(9) COMPUTATIONAL.
    03  SACDDIPU     PICTURE X.
    SKIP3
*      ERROR MESSAGE FOR BAD STATUS CODE
 01  BADSTAT.
    03  A            PICTURE X(6)   VALUE'ERROR '.
    03  RDWR         PICTURE X(4).
    03  B            PICTURE X(27)
                       VALUE 'ING DATABASE, STATUS CODE: '.
    03  STAT         PICTURE XX.
 EJECT

 LINKAGE SECTION.
 01  SPADSECT COPY SPACOBOL.
     EJECT

 PROCEDURE DIVISION USING SPADSECT.
```

```
*****************************************************************
*     MAIN MODULE LOGIC                                         *
*                                                               *
*     CHECK IF FIRST TIME THROUGH.  IF SO, DO FIRST-TIME        *
*     PROCESSING, OTHERWISE MAP THE SCREEN RESPONSE AND FIELDS  *
*     FROM IV SEG INTO WORK AREA.  PROCESS INPUT AND CALL       *
*     AUDITOR.                                                  *
*                                                               *

*****************************************************************

 MAIN-LOGIC.
     IF SPAFIRST = 0 THEN
       PERFORM SET-SPAFIRST-SWITCH THRU SET-SPAFIRST-SWITCH-EXIT
     ELSE
       PERFORM MAP-WORK-AREA THRU MAP-WORK-AREA-EXIT.
     MOVE ECODE TO RETURN-CODE.
     GOBACK.
     EJECT
*****************************************************************
*    PERFORMED ROUTINES                                         *
*                                                               *
*****************************************************************

     SKIP3

*****************************************************************
*                                                               *
*    SET-UP ON FIRST-TIME PROCESSING                            *
*                                                               *
*****************************************************************

 SET-SPAFIRST-SWITCH.
     MOVE 1 TO SPAFIRST.     NOTE INDICATE SECOND TIME THRU.
     MOVE 3 TO ECODE.        NOT SET DISPLAY SCREEN WITH MSG.
     MOVE 'ENTER CLOSE|STOCK INCR|DISBURSE QUANTITYS-P|U'
       TO SPAERMSG.
     IF GOTPD = 'N' THEN
       PERFORM READPD THRU READPD-EXIT.  NOT GET PD SEGMENT.
 SET-SPAFIRST-SWITCH-EXIT.
     EXIT.
     EJECT
```

```
*****************************************************************
*                                                               *
*    MAP SCREEN RESPONSE AND FIELDS FROM IV SEG INTO WORK AREA  *
*                                                               *
*      SECOND TIME: PROCESS INPUT. CALL AUDITOR TO CHECK P|U    *
*                                                               *
*****************************************************************

 MAP-WORK-AREA.
     CALL 'AUDITOR'.
     IF SPARTNCD NOT &GTSYM. 4 THEN
       GO TO OK-AUDIT.       NOTE AUDIT OK.
     MOVE 8 TO ECODE.        NOTE SET RETURN CODE FOR AUDIT FAIL.
     GO TO MAP-WORK-AREA-EXIT.   NOTE RETURN FOR CORRECTION.
 OK-AUDIT.
     MOVE 0 TO SPAFIRST.     NOTE RESET TO FIRST TIME INDICATION.
     MOVE 0 TO OPT.          NOTE MAP INTO MY AREA FROM SPA.
     PERFORM MAP THRU MAP-EXIT.  NOTE CALL MAPPER.
     MOVE 3 TO ECODE.        NOTE REDISPLAY SCREEN & TERMINATE.
     MOVE 'CLOSE/DISBURSE FUNCTION EXECUTED SUCCESSFULLY'
       TO SPAERMSG.
     IF SACDCLOR NOT = 0 THEN
       PERFORM CLOSE-QTY-ENTERED THRU CLOS-QTY-ENTERED-EXIT.
     IF SACDDQTY NOT = 0 THEN
       PERFORM DISBURSE-QTY-ENTERED THRU
         DISBURSE-QTY-ENTERED-EXIT.
     IF SACDCLOR NOT = 0 OR SACDDQTY NOT = 0 THEN
       PERFORM WRITE-RTN THRU WRITE-RTN-EXIT. NOTE UPDATE IV SEG.
 MAP-WORK-AREA-EXIT.
     EXIT.
     EJECT
*****************************************************************
*                                                               *
*    ROUTINE TO UPDATE AND REPLACE IV SEGMENT                   *
*                                                               *
*****************************************************************

 WRITE-RTN.
     MOVE 1 TO OPT.              NOTE MAP TO SEGMENT.
     PERFORM MAP THRU MAP-EXIT.  NOTE CALL MAPPER.
     CALL 'SEGHNDLR' USING IV, HREP.
     IF SPADLIST = SPACES AND SPARTNCD = 0 THEN
       GO TO WRITE-RTN-EXIT.     NOTE REPLACE WAS OK, SO GET OUT.
     MOVE 3 TO ECODE.            NOTE DRIVER TO SHOW ERROR MSG.
     MOVE SPADLIST TO STAT.      NOTE MOVE STATUS CODE TO MSG.
     MOVE 'WRIT' TO RDWR.        NOTE INDICATE WRITE ERROR.
     MOVE BADSTAT TO SPAERMSG.   NOTE MOVE ERROR MESSAGE TO SPA.
 WRITE-RTN-EXIT.
     EXIT.
     EJECT
```

```
*********************************************************************
*                                                                   *
*    ROUTINE TO READ PD SEGMENT                                     *
*                                                                   *
*********************************************************************

 READPD.
     MOVE SPAKEYID TO COKEY.      NOTE KEY FOR PA SEGMENT.
     MOVE '02' TO COKEY-PD.       NOTE CREATE KEY FOR PD SEGMENT
     CALL 'SEGHNDLR' USING PD, GU, COKEY.
     IF SPADLIST NOT = SPACES OR SPARTNCD NOT = 0 THEN
       PERFORM BAD-READPD
     ELSE
       MOVE 'Y' TO GOTPD.         NOTE READPD OK.
 READPD-EXIT.
     EXIT.
     SKIP3
*  BAD RETURN CODE OR STATUS
     SKIP1
 BAD-READPD.
     MOVE SPADLIST TO STAT.       NOTE STATUS CODE TO MSG.
     MOVE 'READ' TO RDWR.         NOTE INDICATE READ.
     MOVE BADSTAT TO SPAERMSG.    NOTE MSG TO SPA.
     EJECT

*********************************************************************
*                                                                   *
*    MAPPING ROUTINE                                                *
*                                                                   *
*********************************************************************

 MAP.
     CALL 'MAPPER' USING MAPRULE, WORKURCD, OPT.
     IF SPARTNCD &GTSYM. 4 THEN
       DISPLAY 'ERROR IN MAPPER'. NOTE BAD RETURN CODE FROM V44.
 MAP-EXIT.
     EXIT.
     EJECT
```

```
*****************************************************************
*                                                               *
*    PROCESS CLOSE QUANTITY                                     *
*                                                               *
*****************************************************************

 CLOSE-QTY-ENTERED.
     SUBTRACT SACDCLOR FROM SAIVONOR.  NOTE DECREMENT ON ORDER.
     IF SAIVONOR &LTSYM. 0 THEN
       MOVE '*** WARNING: NEGATIVE ON-ORDER POSITION ***'
         TO SPAERMSG.             NOTE ON ORDER BECAME NEGATIVE.
*  USE STOCK INCREMENT IF ENTERED, ELSE USE CLOSE QTY AS INCREMENT
     IF SACDCLST NOT = 0 THEN
        ADD SACDCLST TO SAIVSTCK
     ELSE
       ADD SACDCLOR TO SAIVSTCK.
 CLOSE-QTY-ENTERED-EXIT.
     EXIT.
     EJECT.
*****************************************************************
*                                                               *
*    PROCESS DISBURSE QUANTITY                                  *
*                                                               *
*****************************************************************

 DISBURSE-QTY-ENTERED.
     IF SACDDQTY NOT &GTSYM. SAIVSTCK THEN
       GO TO CHK-PLANNED-DISB.      NOTE TOTAL STOCK NOT NEGATIVE.
     MOVE 'DISBURSE NOT DONE-WOULD CAUSE NEG. TOTAL' TO SPAERMSG.
     MOVE ECODE TO RETURN-CODE.     NOTE CAN'T HAVE NEG TOTAL STK.
     GOBACK.

SKIP3

*  INCREMENT PLANNED OR UNPLANNED DISBURSEMENT DEPENDING ON CODE.
 CHK-PLANNED-DISB.
     IF SACDDIPU = 'P' THEN
       ADD SACDDQTY TO SAIVDIPL
     ELSE
       ADD SACDDQTY TO SAIVDIUN.    NOTE INCREMENT DISBURSEMENT.
     SUBTRACT SACDDQTY FROM SAIVSTCK.  NOTE DECREMENT STOCK.
     SKIP3
 DISBURSE-QTY-ENTERED-EXIT.
     EXIT.                          NOTE RETURN TO CALLER
```

COBOL SPA Layout Listing

The following lines represent the COBOL SPA layout, which is in macro library IMSADF.ADFMAC.

```
*****************************************************************
*                                                               *
*       SCRATCH PAD AREA (SPA) DESCRIPTION IMSADF II V2 R2      *
*                                                               *
*****************************************************************
*
 01  SPADSECT.
   03  SPALEGTH        PICTURE 9(4)   COMPUTATIONAL.
   03  FILLER          PICTURE X(4).
   03  SPATRANS        PICTURE X(8).
   03  SPAPROFI.
     05  SPAPKEY.
       07  SPAPROID    PICTURE XX.
       07  SPAPVK      PICTURE X.
     05  SPAPKNXT.
       07  SPAPKNID    PICTURE XX.
       07  SPAPKNV     PICTURE X.
     05  FILLER        PICTURE X(4).
     05  SPAOMSW       PICTURE XX.
     05  FILLER REDEFINES SPAOMSW PICTURE 9(4) COMPUTATIONAL.
       88  SPADBMSW    VALUE 2048.
     05  SPANOTRX      PICTURE 9(4)   COMPUTATIONAL.
     05  SPAPFTRX      PICTURE X(300).
     05  SPAPFTRX-R REDEFINES SPAPFTRX OCCURS 75 TIMES.
       07  SPAPTRXS    PICTURE XX.
       07  SPAPTRXC    PICTURE X.
       07  SPATPTRX    PICTURE X.
         88  SEGMENT-TRX VALUE '0'.
         88  UTILITY-TRX VALUE '1'.
     05  FILLER        PICTURE X(16).
     05  SPASQLCD    PICTURE 9(9) COMPUTATIONAL.
     05  FILLER        PICTURE X(2).

   03  SPAACCTR.
     05  SPASYSID    PICTURE X(4).
     05  SPAULANG    PICTURE X(1).
     05  SPAMFSPF    PICTURE X(2).
     05  FILLER      PICTURE X(1).
     05  SPAMANNO    PICTURE X(6).
     05  SPAUSER     PICTURE X(11).
     05  SPADATE     PICTURE X(5).
     05  SPASIGON    PICTURE X(6).
     05  SPAMODSI    PICTURE X(4).
     05  SPAWHERE    PICTURE 9(4) COMPUTATIONAL.
     05  SPALTERM    PICTURE X(8).
     05  SPAEDATE    PICTURE X.
     05  SPACLUS     PICTURE X(2).
     05  SPAVBITS    PICTURE X.
     05  SPASQLKS    PICTURE X.
     05  SPAWARN     PICTURE X.
     05  SPATBITS    PICTURE X.
     05  SPAPROJ     PICTURE X.
     05  SPAPFK      PICTURE XX.
```

```
  05  SPAGROUP    PICTURE X.
  05  SPABITS     PICTURE X.
  05  SPAITRSZ    PICTURE 9(4) COMPUTATIONAL.
  05  SPACOMLN    PICTURE 9(4) COMPUTATIONAL.
  05  SPALSTLN    PICTURE X.
  05  SPALSTOF    PICTURE X.
  05  SPAFIRST    PICTURE 9(4) COMPUTATIONAL.
  05  SPARTNCD    PICTURE 9(9) COMPUTATIONAL.
    88  DLI-OK    VALUE 0.
    88  GE-STAT   VALUE 4.
    88  DB-ERR    VALUE 8.
    88  BAD-CALL  VALUE 12.
  05  SPASECTX    PICTURE 9(4) COMPUTATIONAL.
    88  NOSECTRX  VALUE 0.
    88  SECOK     VALUE 1.
    88  SECER     VALUE 2.
    88  SEC-OK-ER VALUE 3.
  05  FILLER      PICTURE XX.
  05  SPADLIST    PICTURE XX.

03  SPAPGOPT      PICTURE 9(4) COMPUTATIONAL.
03  SPASWITH        PICTURE XX.
03  SPASWITH-R REDEFINES SPASWITH PICTURE 9(4) COMPUTATIONAL.
  88  SPACHGTR      VALUE ' '.
   * 88  SPACHGTR      HAS A BINARY VALUE OF 32768
  88  SPATSESW      VALUE ' '.
   * 88  SPATSESW      HAS A BINARY VALUE OF 16384
  88  SPAKSEND      VALUE 8192.
  88  SPACHGPG      VALUE 4096.
  88  SPASEGUT      VALUE 2048.
  88  SPANPROT      VALUE 1024.
  88  SPACHGID      VALUE 512.
  88  SPASPUTL      VALUE 256.
  88  SPABDSON      VALUE 128.
  88  SPAMSUTL      VALUE 64.
  88  SPATXTUT      VALUE 32.
  88  SPASAMTX      VALUE 16.
  88  SPAEOFSW      VALUE 8.
  88  SPAROLLC      VALUE 4.
  88  SPACHGM4      VALUE 2.
03  SPATRX.
  05  SPATRXCD      PICTURE X.
  05  SPATRXSG      PICTURE XX.
03  SPACGTRX        PICTURE XXX.
03  SPAMODNM        PICTURE X(8).
03  SPAERMSG        PICTURE X(50).
03  SPATXEBC        PICTURE X(8).
03  SPASHOTR        PICTURE X(8).
03  SPAOPTON        PICTURE X.
03  SPACCODE        PICTURE XX.
03  SPAHLEV         PICTURE X.
03  SPAKEYLG        PICTURE 9(4) COMPUTATIONAL.
03  SPADBKLG        PICTURE 9(4) COMPUTATIONAL.
03  SPASCNCT        PICTURE 9(4) COMPUTATIONAL.
03  SPAITSEG        PICTURE 9(4) COMPUTATIONAL.
03  SPAKSOFF        PICTURE 9(4) COMPUTATIONAL.
03  SPAENDIN        PICTURE 9(9) COMPUTATIONAL.
03  SPANXTAD        PICTURE 9(4) COMPUTATIONAL.
```

```
03  SPARLCNT        PICTURE 9(4) COMPUTATIONAL.
03  SPASTOFF        PICTURE 9(4) COMPUTATIONAL.
03  SPARTOFF        PICTURE 9(4) COMPUTATIONAL.
03  SPAITOFF        PICTURE 9(4) COMPUTATIONAL.
03  SPADBKOF        PICTURE 9(4) COMPUTATIONAL.
03  SPADEVLN        PICTURE X.
03  SPAKEYID        PICTURE X(255).
03  SPAUTILY.
  05  SPAFLDSG      PICTURE X(2000).
```

PL/I Source Listing for SAMPUCD

```
PLIPROC:
    PROC(SPA);
%SKIP (3);
DCL (SPA) FIXED BIN (31);
%/SKIP (3);
                                /*SET UP EXECUTION TIME OPTIONS */
DCL PLIXOPT CHAR(50) VAR INIT('NOCOUNT,NOFLOW,NOREPORT,NOSTAE,NOSPIE')
       STATIC EXTERNAL;
%/SKIP(3)
            /* ASSEMBLER LANGUAGE SUBROUTINES AND ENTRY POINTS */
DCL (AUDITOR,MAPPER,SEGHNDLR) ENTRY OPTIONS(ASSEMBLER);
%SKIP;
DCL MAPSRULE CHAR (2) INIT('M1');     /* NAME FOR MAPPING RULE THAT */
                                      /* CONTROLS MAPPING OF DATA   */
%PAGE;
/*****************************************************************/
/*                                                               */
/*    SCRATCH PAD AREA(SPA) DESCRIPTION FOR IMSADF II V2 R2      */
/*                                                               */
/*****************************************************************/
 DCL SPAPTR  POINTER;              /* SPA POINTER                */
 DCL
   1 SPADSECT  BASED(SPAPTR),
    2 SPALEGTH FIXED BIN(15),/* 1    LENGTH OF SPA               */
    2 FILL1     CHAR(4),     /* 3      RESERVED FOR IMS          */
    2 SPATRANS  CHAR(8),     /* 7      IMS TRANSACTION           */
    2 SPAPROFI, /* CHAR(334)    15     USER PROFILE DESC         */
     3 SPAPKEY,              /* 15     PROFILE SEGMENT KEY       */
      4 SPAPROID  CHAR(2),   /* 15     PROFILE ID                */
      4 SPAPVK    CHAR(1),   /* 17     VARIABLE PORTION OF KEY   */
     3 SPAPKNXT,             /* 18     CONTINUATION SEGMENT KEY  */
      4 SPAPKNID  CHAR(2),   /* 18     PROFILE ID                */
      4 SPAPKNV   CHAR(1),   /* 20     'VARIABLE' PORTION OF KEY */
     3 FILL2     CHAR(4),    /* 21     RESERVED                  */
     3 SPAOMSW,              /* 25     SWITCHES FOR OPTION MENU  */
      4 FILL3     BIT(4),    /*          RESERVED                */
      4 SPADBMSW  BIT(1),    /*       1 = DATA BASE MANAGER      */
      4 FILL4    BIT(11),    /*          RESERVED                */
     3 SPANOTRX FIXED BIN(15),/*27     NUMBER OF TRXS USER ALLOWED */
     3 SPAPFTRX(75),/*CHAR(300) 29     TRX TABLE--UP TO 75 ALLOWED */
      4 SPAPTRXS  CHAR(2),   /*   (29) SEGMENT INVOLVED          */
      4 SPAPTRXC  CHAR(1),   /*   (31) TRX CODE (1-8)            */
      4 SPATPTRX  CHAR(1),   /*   (32) TYPE OF TRANSACTION       */
                             /*    0 = SEGMENT                   */
                             /*    1 = UTILITY                   */
    3 FILLER CHAR(16),      /*   FILL UP TO 344--UNUSED AREA     */
    3 SPASQLCD FIXED BIN(31),/*   SQL RETURN CODE            2.2 */
    2 FILL2B  CHAR(2),       /*        SPARE BYTES              */
    2 SPAACCTR,              /*        ACCOUNTING INFO AREA     */
     3 SPASYSID  CHAR(4),    /*  351   CONTAINS SYSTEM IDENT    */
     3 SPAULANG  CHAR(1),    /*  355   LANGUAGE FOR CONVERSATION */
     3 SPAMFSPF  CHAR(2),    /*  356   PREFIX FOR COMMON SCREENS */
     3 FILL4A    CHAR(1),    /*  358   RESERVED                  */
     3 SPAMANNO  CHAR(6),    /*  359   MAN NUMBER OF USER        */
     3 SPAUSER   CHAR(11),   /*  365   USER WHO HAS SIGNED ON    */
```

```
 3 SPADATE   CHAR(5),    /*  376   DATE USER SIGNED ON         */
 3 SPASIGON  CHAR(6),    /*  381   TIME USER SIGNS ON          */
 3 SPAMODSI  CHAR(4),    /*  387   SIGN ON MOD NAME(4 CHARS)2. 1*/
 3 SPAWHERE FIXED BIN(15),/* 389  OFFSET TO  SQL USER KEY      */
 3 SPALTERM  CHAR(8),    /*  393   LOGICAL TERMINAL NAME       */
 3 SPAEDATE  CHAR(1),    /*  401   E=EUROPEAN DATE     7/81    */
 3 SPACLUS   CHAR(2),    /*  402   ALT CLUS CODE      JR 12/81 */
 3 SPAVBITS  CHAR(1),    /*  404   VARIOUS FLAG BITS           */
  4 SPADBCS   BIT(1),    /*  404   1=USING DBCS                */
 3 SPASQLKS  CHAR(1),    /*  405   USER KEY SEL ROUTINE 3-9 2.2*/
 3 SPAWARN,              /*  406   SQL WARNING BITS         2.2*/
  4 SPAWARN0  BIT(1),    /*  406   WARNING0    1=W          2.2*/
  4 SPAWARN1  BIT(1),    /*  406   WARNING1    1=W          2.2*/
  4 SPAWARN2  BIT(1),    /*  406   WARNING2    1=W          2.2*/
  4 SPAWARN3  BIT(1),    /*  406   WARNING3    1=W          2.2*/
  4 SPAWARN4  BIT(1),    /*  406   WARNING4    1=W          2.2*/
  4 SPAWARN5  BIT(1),    /*  406   WARNING5    1=W          2.2*/
  4 SPAWARN6  BIT(1),    /*  406   WARNING6    1=W          2.2*/
  4 SPAWARN7  BIT(1),    /*  406   WARNING7    1=W          2.2*/
 3 SPATBITS,             /*  407   BITS FOR TWIN PROCESSING 2.2*/
  4 SPAETWIN  BIT(1),    /*  407   1=GE ON TWIN RETRIEVAL   2.2*/
  4 SPARTWIN  BIT(1),    /*  407   1=TWIN RE-RETRIEVAL NEED 2.2*/
  4 SPAFTWIN  BIT(1),    /*  407   1= RETRIEVING FIRST TWIN 2.2*/
  4 SPAPTWIN  BIT(1),    /*  407   1=TWINS BEING PROCESSED  2.2*/
 3 SPAPROJ   CHAR(1),    /*  408   PROJECT INDICATOR           */
 3 SPAPKF    CHAR(2),    /*  409                               */
 3 SPAGROUP  CHAR(1),    /*  411   GROUP NUMBER                */
 3 SPABITS,              /*  412   FLAGS                       */
  4 SPAAUDMG  BIT(1),    /*  412 1= USE AUDIT PRODUCED MSG     */
  4 SPASPR03  BIT(1),    /*  412 1= SPR R.C.3-SAVE SPERMG      */
  4 SPABMP09  BIT(1),    /*  412 1= T09 A BMP & AIOPCB AVAIL2.1*/
  4 SPANOPTN  BIT(1),    /*  412 1= N OPT AN NO MORE ENTRIES   */
  4 SPAAUDER  BIT(1),    /*  412 1= ONLY 1 AUDIT ERROR         */
  4 SPADLIEX  BIT(1),    /*  412 1= DLI EXIT FOR THIS TRXID 2.1*/
  4 SPACOMP   BIT(1),    /*  412 1= COMPOSITE LMOD FOR TRXID2.2*/
  4 SPALIKE   BIT(1),    /*  412 1= DB2 LIKE REQUEST'_ OR %'2.2*/
 3 SPAITRSZ FIXED BIN(15),/* 413   LENGTH OF ITR ENTRIES       */
 3 SPACOMLN FIXED BIN(15),/* 415   LEN OF SPAUTILY  (COMM AREA)*/
 3 SPALSTLN  CHAR(1),     /* 417   CURRENT SEGMENT'S LENGTH    */
 3 SPALSTOF  CHAR(1),     /* 418   CURRENT SEG'S SPAKEYID OFFST*/
 3 SPAFIRST FIXED BIN(15),/* 419   FIRST SWITCH FOR UTILITY RTN*/
 3 SPARTNCD FIXED BIN(31),/* 421   RETURN CODE AREA -UTILITY RT*/
 3 SPASECTX FIXED BIN(15),/* 425   SPECIAL PROC SEC TRX SWITCH.*/
                         /*       0=NO SEC TRX                */
                         /*       1=SEC TRX IF SECOK=YES      */
                         /*       2=SEC TRX IF SECER=YES      */
                         /*       3=SEC TRX IF EITHER=YES     */
 3 FILL8     CHAR(2),    /*  427   RESERVED                    */
 3 SPADLIST  CHAR(2),    /*  429   DL/I STATUS CODE AREA.      */
2 SPAPGOPT FIXED BIN(15),/*  431   CONTAINS OPTION NUMBER FOR  */
                         /*         PROGRAM IN CONTROL TO TAKE */
2 SPASWITH,              /*  433   SWITCHES FOR PROGRAMS       */
 3 SPACHGTR  BIT(1),     /*  1 = NEXT TRX ENTERED BY USER      */
 3 SPATSESW  BIT(1),     /*  1 = INFORMATIONAL MESSAGE FOR     */
                         /*             THE OPTION MENU.       */
 3 SPAKSEND  BIT(1),     /*  1 = END OF KEY SELECTION RULE     */
                         /*             FOUND                  */
 3 SPACHGPG  BIT(1),     /*  1 = INDICATES TO SESSION TERM     */
```

```
                              /*      MODULE JUST TO LOG TO ACCT    */
                              /*      DB AND RETURN TO OPT MENU     */
       3 SPASEGUT  BIT(1),    /*  0 = SEGMENT SELECTION             */
                              /*  1 = UTILTY SELECTION              */
       3 SPANPROT  BIT(1),    /*  1 = DO NOT PROTECT PRIMARY KEY    */
                              /*      FIELDS FOR UTILITY FOR 2ND    */
                              /*      3RD, ETC SEGMENTS IN          */
                              /*      ERROR MODE                    */
       3 SPACHGID  BIT(1),    /*  1 = NEXT KEY ENTERED BY USER      */
       3 SPASPUTL  BIT(1),    /*  1 = UTILITY THROUGH KEY SELECT    */
       3 SPABDSON  BIT(1),    /*  1 = USER SIGNED ON INCORRECTLY    */
                              /*      DISPLAY SIGN ON SCREEN        */
       3 SPAMSUTL  BIT(1),    /*  1 = MSG SENDING REQUEST BY UTILITY*/
       3 SPATXTUT  BIT(1),    /*  1 = TEXT UTITLIY OPTION           */
       3 SPASAMTX  BIT(1),    /*  1 = SAME INPUT TRX CODE-BATCH ONLY*/
       3 SPAEOFSW  BIT(1),    /*  1 = END OF FILE HAS BEEN REACHED  */
       3 SPAROLLC  BIT(1),    /*  1 = ROLL CALL ON COMPARE FAIL     */
       3 SPACHGM4  BIT(1),    /*  1 = MODE 4 FIELD CHNGD, NON-CONV  */
       3 FILL9     BIT(1),    /*       RESERVED                     */
      2 SPATRX,               /*       TRX USER SELECTED            */
       3 SPATRXCD  CHAR(1),   /* 435   TRK CODE - (1-8)             */
       3 SPATRXSG  CHAR(2),   /* 436   TRANSACTION SELECTED         */
      2 SPACGTRX  CHAR(3),    /* 438   CHANGED TRX SPECIFIED BY USER*/
      2 SPAMODNM  CHAR(8),    /* 441   DISPLAY MOD NAME TO BE CALLED*/
                              /*        NEXT - TAKEN FROM INPUT RULE*/
      2 SPAERMSG  CHAR(50),   /* 449   ERROR OR INFORMATION MESSAGE */
                              /*       TO BE PASSED FROM ONE PROGRAM*/
                              /*       TO ANOTHER. SWITCH INDICATES */
      2 SPATXEBC  CHAR(8),    /* 499   TRX CODE IN EBCDIC           */
      2 SPASHOTR  CHAR(8),    /* 507   IMS TRX CODE IN PROGRESS     */
      2 SPAOPTON  CHAR(1),    /* 515   OPT SELECTED FROM OPTION MENU*/
      2 SPACCODE  CHAR(2),    /* 516   CLUSTER CODE SAVED IN KEY SEL*/
      2 SPAHLEV   CHAR(1),    /* 518   HIGHEST LEV AUTH FOR TRXID   */
      2 SPAKEYLG FIXED BIN(15),/* 519   PRIMARY KEY LENGTH FOR SEGMNT*/
      2 SPADBKLG FIXED BIN(15),/* 521   DATA BASE CONCATENATED KEYLEN*/
      2 SPASCNCT FIXED BIN(15),/* 523   SCREEN NUMBER TO BE SHOWN -  */
                              /*       FORMATTER AND DEFORMATTER    */
      2 SPAITSEG FIXED BIN(15),/* 525   ENTRY # OF ITRSEG LAST PROCSD*/
      2 SPAKSOFF FIXED BIN(15),/* 527   REL OFFSET OF KEY SEL INFO   */
      2 SPAENDIN FIXED BIN(31),/* 529   OFFSET OF AUDIT INFO FROM    */
                              /*       START OF INPUT RULE          */
      2 SPANXTAD FIXED BIN(15),/* 533   PTR TO NEXT AVAIL POSITION   */
                              /*       IN RULE - SEGMENT AREA       */
      2 SPARLCNT FIXED BIN(15),/* 535   NUMBER OF ENTRIES IN TABLE   */
                              /* THE FOLLOWING OFFSETS ARE RELATIVE */
                              /* TO THE BEGINNING OF SPAFLDSG       */
      2 SPASTOFF FIXED BIN(15),/* 537   RELATIVE OFFSET TO SPAAUDIT  */
      2 SPARTOFF FIXED BIN(15),/* 539   RELATIVE OFFSET TO SPARLOFF  */
      2 SPAITOFF FIXED BIN(15),/* 541   RELATIVE OFFSET TO SPAINPRL  */
      2 SPADBKOF FIXED BIN(15),/* 543   RELATIVE OFFSET TO SPADBKEY  */
      2 SPADEVLN CHAR(1),     /* 545   NUMBER OF LINES ON SCREEN    */
      2 SPAKEYID CHAR(255),   /* 546   DISPLAYABLE CONCATENATED KEY */
      2 SPAUTILY,             /* 801   COMM AREA  (VARIABLE LENGTH) */
       3 SPAFLDSG CHAR(2000); /* 801     STORAGE AREA FOR SEGMENTS, */
                              /*         FIELD AND INPUT RULES.     */
  DCL CHARSPA(6000) CHAR(1) BASED (SPAPTR); /*CHAR OVERLAY FOR SPA  */
%PAGE;
```

```
 /*************************************************************/
 /*                                                           */
 /*  PARAMETER LIST FOR THE SEGMENT HANDLER                   */
 /*                                                           */
 /*  TWO TO SIX PARAMETERS MUST BE PASSED TO THE SEGMENT      */
 /*  HANDLER INTERFACE.  THE PARMS ARE:                       */
 /*                                                           */
 /*      1 - 2 CHAR ID OF THE SEGMENT DESIRED.                */
 /*      2 - 4 CHAR DL/I FUNCTION TO BE PERFORMED.            */
 /*      3 - KEY OF SEGMENT. (OPTIONAL)                       */
 /*          DEFAULTS TO LAST KEY OF SEGMENT IF NOT GIVEN     */
 /*      4 - KEY OPERATOR. (OPTIONAL)                         */
 /*          DEFAULTS TO 'FE'. THIS PARM IS A TWO CHAR        */
 /*          FIELD THAT SAYS IF THE KEY SUPPLIED IN PARM      */
 /*          3 IS A FULLY CONCATENATED KEY OR JUST THE        */
 /*          KEY OF THE TARGET SEGMENT (F=FULL,P=PARTIAL)     */
 /*          AND ALSO TELLS IF THE SEARCH IS TO BE DONE       */
 /*          FOR EQUAL OR GREATER THAN (E=EQUAL,G=GREATER     */
 /*          THAN).                                           */
 /*      5 - A NUMBER (1-5) THAT INDICATES WHICH USER         */
 /*          DATABASE PCB TO USE. (OPTIONAL) 1=DEFAULT.       */
 /*      6 - SEGMENT INPUT/OUTPUT AREA. (OPTIONAL).           */
 /*          DEFAULTS TO USE THE AREA IN THE SPA.             */
 /*                                                           */
 /*  NOTE: IF A PARM IS GIVEN, THEN ALL THE PRECEEDING PARMS  */
 /*        MUST ALSO BE SPECIFIED.                            */

%SKIP(3);

 /*************************************************************/
 /*                                                           */
 /*   PARAMETER LIST TO INVOKE THE SERVICE ROUTINE DATA MAPPER */
 /*                                                           */
 /*      REGISTER 1 POINTING TO A LIST CONSISTING OF THE      */
 /*      FOLLOWING ADDRESSES ON ENTRY.                        */
 /*                                                           */
 /*        1 SERVICE RTN MAPPING RULE NAME (IN AND OUT OF     */
 /*                                         SPA-SEG RULE)     */
 /*        2 MAPPING AREA IN PROGRAM ADDRESS, (WORKURCD)      */
 /*        3 MAPPING OPTION ADDRESS,                          */
 /*                          1 = MAP INTO SERVICE RTN         */
 /*                          0 = MAP OUT OF SERVICE RTN       */
 /*        A RETURN CODE IS PLACED IN SPARTNCD.               */

%SKIP(3);

 /*************************************************************/
 /*                                                           */
 /*     PARAMETER LIST TO INVOKE THE AUDITOR                  */
 /*                                                           */
 /*        THERE ARE NO PARMS THAT HAVE TO BE SUPPLIED THE    */
 /*        AUDITOR FROM THIS PROGRAM.  ALL THE PARMS FOR THE  */
 /*        AUDITOR ARE SET UP BY THE INTERFACE ROUTINE.       */
 /*                                                           */
 /*        A RETURN CODE IS PLACED IN SPARTNCD.               */
```

```
 %PAGE;
DCL                           /* MAPPED INPUT FIELDS           */
 1 WORKURCD,                  /* DATA MAPPER WORK AREA         */
  2 SACDCLOR FIXED BIN(31),   /* CLOSE QUANTITY-ON ORDER DECR  */
  2 SACDCLST FIXED BIN(31),   /* STOCK INCREMENT               */
  2 SACDDQTY FIXED BIN(31),   /* DISBURSE QUANTITY             */
  2 SAIVONOR FIXED BIN(31),   /* ON ORDER                      */
  2 SAIVSTCK FIXED BIN(31),   /* TOTAL STOCK                   */
  2 SAIVDIPL FIXED BIN(31),   /* DISBURSE QTY PLANNED          */
  2 SAIVDIUN FIXED BIN(31),   /* DISBURSE QTY UNPLANNED        */
  2 SACDDIPU CHAR(1);         /* P OR U PLANNED|UNPLANNED DISB */
DCL
 1 BADSTAT,                   /* ERROR MSG FOR BAD STATUS CODE */
  2 A CHAR(06) INIT('ERROR '),
  2 RDWR CHAR(4),             /* 'READ' OR 'WRIT'              */
  2 B CHAR(27) INIT('ING DATABASE, STATUS CODE: '),
  2 STAT CHAR(2);             /* STATUS CODE                   */
DCL BADSTATC CHAR(39) DEFINED BADSTAT; /*CHAR OVERLAY FOR ASSIGN */
DCL
  COKEY CHAR(50) INIT (' ');  /* CONCATENATED KEY BUILD AREA   */
    DCL
 OPT FIXED BIN(31);           /* OPTION FOR DATA MAPPER        */
DCL
 GOTPD CHAR(1) INIT('N'),     /*YES-NO INDICATOR FOR PD READ   */
 ECODE FIXED BIN(31) INIT(0); /* CODE TO PASS TO DRIVER        */

%PAGE;

/*****************************************************************/
/*                                                               */
/* MISCELLANEOUS INITIALIZATION                                  */
/*                                                               */
/*****************************************************************/
                              /*                                 */
    SPAPTR=ADDR(SPA);         /* SET UP POINTERS                 */

%PAGE;

/*****************************************************************/
/*                                                               */
/* MAIN MODULE LOGIC                                             */
/*                                                               */
/*****************************************************************/

    IF SPAFIRST=0 THEN        /* FIRST TIME THRU?                */
 DO;
   SPAFIRST=1;                /* INDICATE SECOND TIME THRU       */
   ECODE=3;                   /* SET-DISPLAY SCREEN WITH MSG     */
   SPAERMSG='ENTER CLOSE|STOCK INCR|DISBURSE QUANTITY -P|U';
   IF GOTPD = 'N' THEN        /*CHECK IF PD SEGMENT READ YET     */
        CALL READPD;          /* GET PD SEGMENT                  */
 END;                         /* END FIRST TIME THRU             */
```

```
   /*************************************************************/
   /*                                                           */
   /* MAP SCREEN RESPONSE AND FIELDS FROM IV SEG INTO WORK AREA*/
   /*                                                           */
   /*************************************************************/

   ELSE
DO;                           /* SECOND TIME: PROCESS INPUT    */
                              /* CALL THE AUDITOR TO CHK P|U   */
  CALL AUDITOR;
  IF SPARTNCD > 4 THEN        /*RC > 4 => AUDIT FAIL           */
     DO;
        ECODE=8;              /*SET RETURN CODE FOR AUDIT FAIL */
        CALL PLIRETC(ECODE);   /*RETURN TO LET USER CORRECT IT */
        RETURN;
     END;
  SPAFIRST=0;                 /* RESET TO FIRST TIME INDICATION*/
  OPT=0;                      /* MAP INTO MY AREA FROM SPA     */
  CALL MAP;                   /* CALL MAPPER                   */
  ECODE=3;                    /* REDISPLAY SCREEN & TERMINATE  */
  SPAERMSG='CLOSE/DISBURSE FUNCTION EXECUTED SUCCESSFULLY';
  IF SACDCLOR¬=0 THEN         /* HAS CLOSE QTY BEEN ENTERED?   */
    DO;                       /* YES                           */
      SAIVONOR=SAIVONOR-SACDCLOR;/* DECREMENT ON ORDER         */
      IF SAIVONOR<0 THEN      /* HAS IT BECOME NEGATIVE?       */
        SPAERMSG='*** WARNING: NEGATIVE ON-ORDER POSITION ***';
      IF SACDCLST¬=0 THEN     /* WAS STOCK INCREMENT ENTERED?  */
        SAIVSTCK=SAIVSTCK+SACDCLST;/* YES - INCREMENT STOCK */
      ELSE
        SAIVSTCK=SAIVSTCK+SACDCLOR;/* NO - USE CLOSE QTY AS INCR*/
    END;
  IF SACDDQTY¬=0 THEN         /* WAS DISBURSE QTY ENTERED?     */
    DO;                       /* YES                           */
      IF SACDDQTY > SAIVSTCK THEN
         DO; SPAERMSG='DISBURSE NOT DONE-WOULD CAUSE NEG. TOTAL';
             CALL PLIRETC(ECODE);  /*CAN'T HAVE NEG. TOTAL STCK*/
             RETURN;
         END;
      IF SACDDIPU='P' THEN    /* PLANNED DISBURSEMENT?         */
        SAIVDIPL=SAIVDIPL+SACDDQTY;/* YES-INCREMENT PLANNED DISB*/
      ELSE
        SAIVDIUN=SAIVDIUN+SACDDQTY;/*NO-INCREMENT UNPLANNED DISB*/
      SAIVSTCK=SAIVSTCK-SACDDQTY;/* DECREMENT STOCK            */
    END;
  IF SACDCLOR¬=0 | SACDDQTY¬=0 THEN/* WAS CLOSE OR DISB ENTERED*/
     CALL WRITE;              /* YES-UPDATE IV SEGMENT         */
END;
   CALL PLIRETC(ECODE);
   RETURN;

%PAGE;
```

```
/*****************************************************************/
/*                                                               */
/* PROC TO UPDATE AND REPLACE IV SEGMENT                         */
/*                                                               */
/*****************************************************************/

 WRITE: PROC;
    OPT=1;                     /* MAP TO SEGMENT               */
    CALL MAP;                  /* CALL MAPPER                  */
    CALL SEGHNDLR('IV','HREP');/* REPLACE IV SEG AFTER GHU DONE.*/
    IF SPADLIST ¬='  ' | SPARTNCD ¬= 0 THEN
 DO;                           /* BAD REPLACE                  */
   ECODE=3;                    /* TELL DRIVER TO SHOW ERROR MSG */
   STAT=SPADLIST;              /* MOVE STATUS CODE TO MSG      */
   RDWR='WRIT';                /* INDICATE WRITE ERROR         */
   SPAERMSG=BADSTATC;          /* MOVE ERROR MSG TO SPA        */
 END;                          /* END REPLACE ERROR            */
    END;                       /* END UPDATE AND REPLACE DS    */

%PAGE;

/*****************************************************************/
/*                                                               */
/* PROC TO READ PD SEGMENT                                       */
/*                                                               */
/*****************************************************************/

 READPD:
    PROC ;
    COKEY=SPAKEYID;            /* KEY FOR PA SEGMENT           */
           /* SPAKEYID HAS THE FULLY CONCATENATED KEY OF SEGS   */
           /* LOADED BECAUSE OF KEY SELECTION..                 */
    SUBSTR(COKEY,18,2)='02';   /* KEY FOR PD SEGMENT           */
    CALL SEGHNDLR('PD','GU',COKEY); /* GET UNIQUE FOR PD SEGMENT*/
   IF SPADLIST¬='  ' | SPARTNCD ¬= 0 THEN /*CHECK FOR BAD STATUS*/
     DO;                                  /*OR BAD RETURN CODE   */
       STAT=SPADLIST;          /* STATUS CODE TO MSG           */
       RDWR='READ';            /* INDICATE READ                */
       SPAERMSG=BADSTATC;      /* MOVE MSG TO SPA              */
     END;
   ELSE
        GOTPD='Y';             /* SET PD SEG READ INDICATOR=YES */
    END;                       /* END READ DS SEGMENT LOGIC    */

%PAGE;

/*****************************************************************/
/*                                                               */
/* MAPPING ROUTINE                                               */
/*                                                               */
/*****************************************************************/

 MAP:
    PROC ;
   CALL MAPPER(MAPSRULE,WORKURCD,OPT); /*GET DATA*/
    IF SPARTNCD > 4 THEN       /* BAD RETURN CODE FROM V44      */
 DISPLAY ('ERROR IN MAPPER');  /* GET OUT                      */
    END;                       /* END MAPPING ROUTINE          */
```

```
END;                    /* END SAMPUCD                  */
```

Rule Generator Requirements to Create a Load Module

A full description of the Rule Generator operands can be found in the *IMSADF II IMS Application Development Facility II Version 2 Release 2 Application Development Reference*.

To link-edit the special processing routine with a copy of the conversational driver base load module, a Rule Generator option is provided.

Compile and link your special processing routine module in IMSADF.RULLIB, which must be in the SYSLIB concatenation. The Segment Handler rules and Mapping rules to be used by the special processing routine must exist in load libraries pointed to by SYSLIB, as must the IMSADF II base load module (MFC1TCT) and the PL/I or COBOL library, as appropriate. The ????G procedure is used to execute the Rule Generator.

Layout of Scratch Pad Area in Assembler (SPAASM)

```
*/*****************************************************************/
*/*                                                               */
*/*    SCRATCH PAD AREA(SPA) DESCRIPTION FOR IMSADF II V2 R2      */
*/*                                                               */
*/*****************************************************************/
SPADSECT DSECT
SPALEGTH DS    H            /* 0     LENGTH OF SPA                */
         DS    CL4          /* 2     RESERVED FOR IMS             */
SPATRANS DS    CL8          /* 6     IMS TRANSACTION              */
SPAPROFI DS    CL334        /* 14    USER PROFILE DESC            */
         ORG   SPAPROFI
SPAPKEY  DS    0CL3         /* 14    PROFILE SEGMENT KEY          */
SPAPROID DS    CL2          /* 14    PROFILE ID                   */
SPAPVK   DS    CL1          /* 16    VARIABLE PORTION OF KEY      */
SPAPKNXT DS    0CL3         /* 17    CONTINUATION SEGMENT KEY     */
SPAPKNID DS    CL2          /* 17    PROFILE ID                   */
SPAPKNV  DS    CL1          /* 19    'VARIABLE' PORTION OF KEY    */
*        DS    CL4          /* 20    RESERVED              PL69275 */
SPACAOFS DS    H            /* 20    OFFSET TO SQL ERR MSG PL69275 */
         DS    CL2          /* 22    RESERVED              PL69275 */
SPAOMSW  DS    CL2          /* 24    SWITCHES FOR OPTION MENU     */
SPADBMSW EQU   X'08'        /*       1 = DATA BASE MANAGER        */
SPANOTRX DS    H            /* 26    NUMBER OF TRXS USER ALLOWED  */
SPAPFTRX DS    CL300        /* 28    TRX TABLE                    */
         ORG   SPAPFTRX
SPAPTRXS DS    CL2          /* 28    SEGMENT INVOLVED             */
SPAPTRXC DS    C            /* 30    TRX CODE (1-8)               */
SPAPTRX  DS    C            /* 31    TYPE OF TRANSACTION          */
*                           /*       C'0' = SEGMENT               */
*                           /*       C'1' = UTILITY               */
         ORG   SPAPFTRX+300 /* 328   LEAVES ROOM FOR 75 TRANSACTION*/
SPANOHLP DS    CL2          /* 328   NEW '?' HELP FLAG   PL64860  */
*  PN02024                  /*       AUDIT SWITCHES WILL  PL69230 */
*  PN02024                  /*       BE SAVED AS BIT      PL69230 */
*  PN02024                  /*       SWITCHES.            PL69230 */
*  PN02024                  /*       ONLY THE FIRST 9     PL69230 */
*  PN02024                  /*       BITS ARE USED.       PL69230 */
SPAASWCH DS    CL2          /* 330   SAVED AUDIT SWITCHES PL69230 */
*        DS    CL16         /* 328   UNUSED CHANGED BY PL64860    */
```

```
*        DS    CL14         /* 330    UNUSED              PL64860   */
* ABOVE LINE COMMENTED OUT BY                            PL69230   */
         DS    CL6          /* 332                        PN02024   */
SPASVPTH DS    CL6          /* 338 SAVE PATH NAME TO SET  PN02024   */
*                           /*     SELMADE                PN02024   */
* THE FOLLOWING LINE COMMENTED OUT BY PN02024             PN02024   */
*        DS    CL12         /* 332    UNUSED              PL69230   */
SPASQLCD DS    F            /* 344    SQL RETURN CODE           2.2 */
*        DS    CL2          /* 348    UNUSED              PN02024   */
SPAERHLP DS    CL2          /* 348    PREAUDIT RETURN CODEPN02024   */
SPAACCTR DS    CL80         /* 350    ACCOUNTING INFO AREA          */
         ORG   SPAACCTR
SPASYSID DS    CL4          /* 350    CONTAINS SYSTEM IDENT         */
SPAULANG DS    CL1          /* 354    LANGUAGE FOR CONVERSATION     */
SPAMFSPF DS    CL2          /* 355    PREFIX FOR COMMON SCREENS     */
*        DS    CL1          /* 357    RESERVED             PN02024  */
SPACMPR  DS    CL1          /* 357    SPA REDUCE INDICATOR PN02024  */
SPAMANNO DS    CL6          /* 358    MAN NUMBER OF USER            */
SPAUSER  DS    CL11         /* 364    USER WHO HAS SIGNED ON        */
SPADATE  DS    CL5          /* 375    DATE USER SIGNED ON           */
SPASIGON DS    CL6          /* 380    TIME USER SIGNS ON            */
SPAMODSI DS    CL4          /* 386    SIGN ON MOD NAME(4 CHARS)     */
SPAWHERE DS    H            /* 388    OFFSET TO SQL USER KEY    2.2 */
SPALTERM DS    CL8          /* 392    LOGICAL TERMINAL NAME         */
SPAEDATE DS    CL1          /* 400    E=EUROPEAN DATE        7/81   */
SPACLUS  DS    CL2          /* 401    ALT CLUS. CODE     JR 12/81   */
SPAVBITS DS    CL1          /* 403    VARIOUS FLAG BITS      2.21503*/
SPADBCS  EQU   X'80'        /* 403    1=USING DBCS           2.21503*/
SPADCERR EQU   X'40'        /* 403    1=DATACOMP ERROR       PL11343*/
SPACOMPR EQU   X'20'        /* 403    COMPRESS SPA SWITCH    PN02024*/
SPAMODCH EQU   X'10'        /* 403    MODE CHANGE - FOR TWINSPN02024*/
SPAPCBCH EQU   X'08'        /* 403                           PN02024*/
SPASTX   EQU   X'04'        /* 403                           PN02024*/
SPASQLKS DS    CL1          /* 404    USER KEY SEL RTN  3-9     2.2 */
SPAWARN  DS    C            /* 405    SWITCHES FOR PROGRAMS         */
SPAWARN0 EQU   X'80'        /*        80-WARNING0     1=W       2.2 */
SPAWARN1 EQU   X'40'        /*        80-WARNING1     1=W       2.2 */
SPAWARN2 EQU   X'20'        /*        80-WARNING2     1=W       2.2 */
SPAWARN3 EQU   X'10'        /*        80-WARNING3     1=W       2.2 */
SPAWARN4 EQU   X'08'        /*        80-WARNING4     1=W       2.2 */
SPAWARN5 EQU   X'04'        /*        80-WARNING5     1=W       2.2 */
SPAWARN6 EQU   X'02'        /*        80-WARNING6     1=W       2.2 */
SPAWARN7 EQU   X'01'        /*        80-WARNING7     1=W       2.2 */
SPATBITS DS    C            /* 406    BITS FOR TWIN PROCESSING  2.2 */
SPAETWIN EQU   X'80'        /*        1=GE ON TWIN RETRIEVE      2.2*/
SPARTWIN EQU   X'40'        /*        1=RE-RETRIEVE NEEDED       2.2*/
SPAFTWIN EQU   X'20'        /*        1=RETRIEVE FIRST TWIN      2.2*/
SPAPTWIN EQU   X'10'        /*        1=TWINS BEING PROCESSED    2.2*/
SPAHELP  EQU   X'08'        /*        1=HELP FACILITY       BA48532 */
SPAA4BIT EQU   X'04'        /*        1=A4 STATUS CODE      PP48928 */
SPASVHLP EQU   X'02'        /*        1=SDS MESSAGE SAVED   PL06343 */
SPAMOPT  EQU   X'01'        /*        1='M' OPT FLAG V54    PL07153 */
SPAPROJ  DS    CL1          /* 407    PROJECT INDICATOR             */
SPAPFK   DS    CL2          /* 408    RESERVED                      */
SPAGROUP DS    CL1          /* 410    GROUP NUMBER                  */
SPABITS  DS    CL1          /* 411                                  */
SPAAUDMG EQU   X'80'        /*        USE AUDIT PRODUCED MESSAGE    */
SPASPR03 EQU   X'40'        /*        SPR SET RETRURN CODE = 3      */
```

```
SPABMP09 EQU   X'20'       /*        BATCH DRIVER RUNNING AS BMP   */
SPANOPTN EQU   X'10'       /*        N OPT ENTERED & END OF SEGS   */
SPAAUDER EQU   X'08'       /*        ONLY 1 ERROR MSG FOR SEGD     */
SPADLIEX EQU   X'04'       /*        DLI EXIT AVAILABLE FOR TRXID  */
SPACOMP  EQU   X'02'       /*        COMPOSIT LMOD TRY FOR TRXID2.2*/
SPALIKE  EQU   X'01'       /*        DB2 LIKE REQUEST          2.2 */
SPAITRSZ DS    H           /* 412    LENGTH OF ITR ENTRIES         */
SPACOMLN DS    H           /* 414    LENGTH OF SPAUTILY (COMM AREA)*/
SPALSTLN DS    X           /* 416    CURRENT SEGMENT'S LENGTH      */
SPALSTOF DS    X           /* 417    CURRENT SEG'S SPAKEYID OFFSET */
SPAFIRST DS    H           /* 418    FIRST SWITCH FOR UTILITY RTN  */
SPARTNCD DS    F           /* 420    RETURN CODE AREA -UTILITY RTN */
SPASECTX DS    H           /* 424    SPEC PROC SEC TRX SWITCH......*/
*                          /*        0=GEN NO SEC TRX              */
*                          /*        1=GEN SEC TRX IF SECOK=YES    */
*                          /*        2=GEN SEC TRX IF SECER=YES    */
*                          /*        3=GEN SEC TRX IF EITHER=YES   */
SPAAUDTB DS    H           /* 426    CUR ERR SEQ TABLE OSET PN02024*/
*        DS    CL2         /* 426    RESERVED              PN02024 */
SPADLIST DS    CL2         /* 428    DLI STATUS AREA FOR SPIR......*/
SPAPGOPT DS    H           /* 430    CONTAINS OPTION NUMBER FOR    */
*                          /*        PROGRAM IN CONTROL TO TAKE    */
SPASWIT1 DS    C           /* 432    SWITCHES FOR PROGRAMS         */
SPACHGTR EQU   X'80'       /*        80-NEXT TRX ENTERED BY USER   */
SPATSESW EQU   X'40'       /*        40-INFORMATIONAL MESSAGE FOR  */
*                          /*             THE OPTION MENU.         */
SPAKSEND EQU   X'20'       /*        20-END OF KEY SELECTION RULE  */
*                          /*             FOUND                    */
SPACHGPG EQU   X'10'       /*        10-INDICATES TO SESSION TERM  */
*                          /*           MODULE JUST TO LOG TO ACCT */
*                          /*           DB AND RETURN TO OPT MENU  */
SPASEGUT EQU   X'08'       /*        08-   0 - SEGMENT SELECTION   */
*                          /*              1 - UTILTY SELECTION    */
SPANPROT EQU   X'04'       /*        04-DO NOT PROTECT PRIMARY KEY */
*                          /*            FIELDS FOR UTILITY FOR 2ND*/
*                          /*            3RD, ETC SEGMENTS IN      */
*                          /*            ERROR MODE                */
SPACHGID EQU   X'02'       /*        02-NEXT KEY ENTERED BY USER   */
SPASPUTL EQU   X'01'       /*        01-UTILITY THROUGH KEY SELECT */
SPASWIT2 DS    C           /* 433    SWITCHES FOR PROGRAMS SWITCH 2*/
SPABDSON EQU   X'80'       /*        80- USER SIGNED ON INCORRECTLY*/
*                          /*            DISPLAY SIGN ON SCREEN    */
SPAMSUTL EQU   X'40'       /*        40- MSG SENDING REQ BY UTILITY*/
SPATXTUT EQU   X'20'       /*        20- TEXT UTITLIY OPTION       */
SPASAMTX EQU   X'10'       /*        10-SAME INPUT TRX CODE-BATCH  */
SPAEOFSW EQU   X'08'       /*        08-END OF FILE HAS BEEN REACHD*/
SPACHGM4 EQU   X'02'       /*        02-MODE 4 FIELD CHANGED.NONCON*/
SPAKSFND EQU   X'01'       /*        1-KEY SEL FUNC ENTERED PN02024*/
SPATRX   DS    CL3         /* 434    TRX USER SELECTED             */
         ORG   SPATRX
SPATRXCD DS    CL1         /* 434    TRK CODE - (1-8)              */
SPATRXSG DS    CL2         /* 435    TRANSACTION SELECTED          */
SPACGTRX DS    CL3         /* 437    CHANGED TRX SPECIFIED BY USER */
SPAMODNM DS    CL8         /* 440    DISPLAY MOD NAME TO BE CALLED */
*                          /*          NEXT - TAKEN FROM INPUT RULE*/
SPAERMSG DS    CL50        /* 448    ERROR OR INFORMATIONAL MESSAGE*/
*                          /*         TO BE PASSED FROM ONE PROGRAM*/
*                          /*         TO ANOTHER. SWITCH INDICATES */
```

```
SPATXEBC DS    CL8         /* 498    TRX CODE IN EBCDIC            */
SPASHOTR DS    CL8         /* 506    IMS TRX CODE IN PROGRESS      */
SPAOPTON DS    CL1         /* 514    OPT SELECTED FROM OPTION MENU */
SPACCODE DS    CL2         /* 515    CLUSTER CODE FOR IMS TRANSACT */
SPAHLEV  DS    CL1         /* 517    HIGHEST AUTHORITY FOR TRXID   */
SPAKEYLG DS    H           /* 518    DISP CONCATENATED KEY LENGTH  */
SPADBKLG DS    H           /* 520    DB CONCAT KEY LEN FOR KEY SEL */
SPASCNCT DS    H           /* 522    SCREEN NUMBER TO BE SHOWN -   */
*                          /*         FORMATTER AND DEFORMATTER    */
SPAITSEG DS    H           /* 524    ENT # OF ITRSEG LAST PROCESSED*/
SPAKSOFF DS    H           /* 526    CURR KEY SELECT       PN02024 */
*        DS    CL2         /* 526    RESERVED              PN02024 */
SPAENDIN DS    F           /* 528    OFFSET OF AUDIT INFO FROM     */
*                          /*         START OF INPUT RULE          */
SPANXTAD DS    H           /* 532    PTR TO NEXT AVAILABLE POSITION*/
*                          /*         RELATIVE TO BEG OF SPAFLDSG  */
SPARLCNT DS    H           /* 534    NUMBER OF ENTRIES IN TABLE    */
SPASTOFF DS    H           /* 536    RELATIVE OFFSET TO SPAAUDIT TB*/
SPARTOFF DS    H           /* 538    RELATIVE OFFSET TO SPARLOFF TB*/
SPAITOFF DS    H           /* 540    RELATIVE OFFSET TO SPAINPRL TB*/
SPADBKOF DS    H           /* 542    RELATIVE OFFSET TO SPADBKEY   */
SPADEVLN DS    CL1         /* 544    NUMBER OF LINES ON SCREEN     */
SPAKEYID DS    CL255       /* 545    KEY FOR SEGMENT HANDLER       */
SPAUTILY EQU   *           /* 800    COMM AREA (VARIABLE LENGTH)   */
SPAFLDSG EQU   *           /* 800    STORAGE AREA FOR              */
*                          /*         SEGMENTS, FIELD, INPUT RULES */
*                          /*         SEGMENT TABLES               */
```

Appendix C. DL/I Space Allocation Calculations

This appendix will help you calculate the direct-access storage requirements for IMASADF II databases when your access method is DL/I.

The appendix contains a blank form to help you do the calculations. Input for the form is as follows:

- Database being reviewed.
- Physical block size being used.
- Storage device type.
- Number of physical blocks per track.
- Estimate of the number of segments for each segment type in the database. This consists of two parts: the actual number of segments distributed with the system and the number of segments an installation expects to add to the database.
- Length of each segment type (including prefix).

Outputs to be used in DBD generation and space allocation are:

- Number of root anchor points (RAPs).
- Number of blocks to be allocated to the root addressable area (BLKS).
- Number of tracks to be in the primary allocation for the data set. This is the number required to accommodate the number of blocks to be allocated to the root addressable area.

This output can be used to modify the:

- Database space allocations in job ADFAALOC of the installation procedure, and/or
- DBDs in the IMSADF.DBDSRC library before running job ADFDBD of the installation procedure

DATA BASE: BLKSIZE: DEVICE:

BLOCKS PER TRACK:

	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8
SEG ID:								
SEG LEN:								
ADF SEGS:								
YOUR SGS:								
ADF+YOUR:								
(A+Y)*LEN:								

```
SUMMATION OF (A+Y)*LEN  =  BR (Byte requirement)

         BR/#ROOT SEGS  =  AV (Average database record)

            BLKSIZE/AV  =  RAPS (Root anchor points)
(ROUNDED UP TO INTEGER)

    BLKSIZE-4-(4*RAPS)  =  EBS (Effective block size)

 (BYTE REQUIRE/.8)/EBS  =  B1 (BLKS for 80% density)

                   B1  =  BLKS (Number of blocks suggested for
                          the root addressable portion
                          of the database--third parameter
                          of RMNAME on the DBD
                          statement in DBDGEN)

BLKS / # BLKS PER TRACK  =  TRKS (Number of tracks of
                          primary allocation for the database.
                          Note: This is the root
                          addressable area, and some
                          secondary allocation should
                          be given for possible overflow.)
```

Notes:

1. SEG ID is the two-character IMSADF II segment identifier.
2. SEG LEN is the segment length.
3. ADF SEGS is the number of segments of the SEG ID type that were distributed with the IMSADF II system.
4. YOUR SGS is the estimated number of this type of segment that your installation adds to the IMSADF II databases.
5. ADF+YOUR or (A+Y) is the sum of ADF SEGS and YOUR SGS for the SEG ID.

DATA BASE: AUDIT BLKSIZE: 4096 DEVICE: 3390

BLOCKS PER TRACK: 13

	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7
SEG ID:	GF	AA	DA	FA	DF	MA	DM
SEG LEN:	34	38	34	38	34	38	34
ADF SEGS:	1			1	1		
YOUR SGS:							
ADF+YOUR:	1082	808	1218	4500	6755	535	808
(A+Y)*LGH:	36788	30704	41412	171000	229670	20330	27472

SUMMATION OF (A+Y)*LGH = 557,376 = BR (Byte requirement)

BR/#ROOT SEGS = 515 = AV (Average database record)
(# ROOTS = # GF = 1082)

BLKSIZE/AV = 8 = RAPS (Root anchor points)
(ROUNDED UP TO INTEGER)

BLKSIZE-4-(4*RAPS) = 4,060 = EBS (Effective block size)

(BYTE REQUIRE/.8)/EBS = 171 = B1 (BLKS for 80% packing density)

B1 = 171 = BLKS (Number of blocks suggested for the database--third parameter of RMNAME on the DBD statement in DBDGEN)

BLKS/# BLKS PER TRACK = 14 = TRKS (Number of tracks of primary allocation for the database.)
Note: This is the root addressable area, and some secondary allocation should be given for possible overflow.

DATA BASE: MESSAGE BLKSIZE: 4096 DEVICE: 3390

BLOCKS PER TRACK: 13

	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8
SEG ID:	AH	AR	HD	SY	SD	LT	UH	US
SEG LEN:	92	84	92	84	92	84	92	162
ADF SEGS:			116	142				
YOUR SGS:								
ADF+YOUR:	300	450	700	2725	104	75	400	800
(A+Y)*LGH:	27600	37800	64400	228900	9568	6300	36800	129600

SUMMATION OF (A+Y)*LGH = 540,968 = BR (Byte requirement)

BR/#ROOT SEGS = 360 = AV (Average database record)
(# ROOTS = # GF = 1503)

BLKSIZE/AV = 11 = RAPS (Root anchor points)
(ROUNDED UP TO INTEGER)

BLKSIZE-4-(4*RAPS) = 4,048 = EBS (Effective block size)

(BYTE REQUIRE/.8)/EBS = 167 = B1 (BLKS for 80% packing density)

B1 = 167 = BLKS (Number of blocks suggested
for the database--third
parameter of RMNAME on the
DBD statement in DBDGEN)

BLKS/# BLKS PER TRACK = 12 = TRKS (Number of tracks of
primary allocation for the
database.)
NOTE: This is
the root addressable area,
and some secondary
allocation should be given
for possible overflow.

DATA BASE: SIGN-ON PROFILE BLKSIZE: 4096 DEVICE: 3390

BLOCKS PER TRACK: 13

	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8
SEG ID:	PG	SR	PR	MG				
SEG LEN:	62	50	340	162				
ADF SEGS:	2	2	2					
YOUR SGS:								
ADF+YOUR:	136	1079	260	208				
(A+Y)*LGH:	8432	53950	88400	33696				

DATA BASE: SIGN-ON PROFILE BLKSIZE: 4096 DEVICE: 3390

BLOCKS PER TRACK: 13

```
 SUMMATION OF (A+Y)*LGH  =  184,478  =  BR (Byte requirement)

          BR/#ROOT SEGS  =  1,356  =  AV (Average database record)
(# ROOTS = # GF = 136)

             BLKSIZE/AV  =  4  =  RAPS (Root anchor points)
(ROUNDED UP TO INTEGER)

     BLKSIZE-4-(4*RAPS)  =  4,076  =  EBS (Effective block size)

  (BYTE REQUIRE/.8)/EBS  =  57  =  B1 (BLKS for 80% packing density)

                     B1  =  57  =  BLKS (Number of blocks suggested
                               for the database--third
                               parameter of RMNAME on the
                               DBD statement in DBDGEN)

  BLKS/# BLKS PER TRACK  =  5  =  TRKS (Number of tracks of
                               primary allocation for
                               the database.)
                               Note: This is the root
                               addressable area, and
                               some secondary allocation
                               should be given for
                               possible overflow.
```

Appendix D. SPA Size Calculations

This appendix shows you how to calculate the:

- Amount of working storage required for various IMSADF II transactions
- Size of the SPA if work area reduction is being used

SPA Calculations for IMSADF II Transactions

Each IMSADF II transaction requires working storage. Non-conversational and batch transactions use working storage directly; conversational transactions use it after the IMS SPA or the IMSADF II HDAM Work database data has been placed in working storage. Figure 88 and Figure 89 show approximately how much working storage (including SPA or HDAM size) is required for various IMSADF II transactions. SIZE assumes no Communication Area is required at sign-on (`COMMLEN=0` was specified).

Figure 88. SPA Requirements for IMSADF II Transactions

TRX TYPE	SIZE
MFC1T01	2,100 bytes
MFC1T02	2,100 bytes
MFC1T03	2,100 bytes
MFC1V01	2,000 bytes
MFC1V05	2,000 bytes
MFC1V06	2,000 bytes
MFC1V07	2,000 bytes
Text Utility	4,000 bytes

Figure 89. SPA Requirements Formula

TRX TYPE	SIZE
Standard Segment and Special Processing	FIXED + VAR + ITR + SLRS + KEYS + SEGS + SAVED + SELECT + USER + IDS

where:

FIXED = 808

VAR = 16 x number of segments

ITR = size of ITR (see linkage editor output)

SLRS = sum of SLRS (see linkage editor output)

KEYS = sum of key lengths for each segment + 12 per segment

SEGS = sum of size of segments (all types)

SAVED = sum of data saved for comparison before update (conversational only)

SELECT = 220 + (SKSEGS x maximum concatenated key length)

This is information for Secondary Key Selection (conversational only). SKSEGS is the Rule Generator parameter specified for the segment on which secondary selection is being done.

USER = Length of the user communication area

(See COMMLEN keyword of GENERATE statement and DEFADF macro.)

IDS = Length of data area identifier within the SPA:

32 + 8 x number of segment IDs

SPA Work Area Reduction Calculations

The following formula is for calculating, on a transaction ID (TRXID) basis, the size of the SPA if the work area reduction option (WRKREDU) is used.

Figure 90. SPA WRKAREA Requirements Formula Using SPA Reduction

TRX TYPE	SIZE in BYTES
Standard Segment and Special Processing	CALCSIZE - 27 - REDUCTION - ITREND

where:

CALCSIZE Size calculated using Figure 89.

REDUCTION Total amount of space saved for **all** applicable items in the following ITEM TABLE. The amount of space saved for each item must be calculated and these individual amounts summed.

The amount of space saved is calculated by following the steps below for **each** item in the ITEM TABLE.

See SIZE REDUCTION in Figure 91.

1. Using the Rule Generator output, count the number of occurrences of the item.
2. Multiply the number of occurrences by the size reduction for that item.

ITREND The number of bytes in the ITR, beginning with the first AUDGP entry, inclusive of the TRXNAME entry.

Figure 91. ITEM TABLE of Size Reductions

ITEM	SIZE REDUCTION A	SIZE REDUCTION B
ITRMOD	27	23
ITRPATH	27	23
ITRSEG	26	22
ITRFLD	25	22
SLR	38	38
FLDRL (SLR field entry)	15	15

It is **not** necessary to do the preceding calculations to determine the reduced SPA size. The easiest way to determine the size of a reduced SPA is to run a BTS test and read the length of the SPA from BTS output.

The savings in space, as calculated using Figure 90, is approximate and can vary slightly during an IMSADF II transaction.

To obtain the information needed to do the calculations, run the Rule Generator with the `ASMR=YES` option. The items in ITEM TABLE are easy to count, and the size of the rule entries can be obtained from the left side of the Assembler listing.

For SIZE REDUCTION

There are two columns in Figure 91 because the ITR entries can be in one of two sizes, depending on whether extended attributes are used (for color or DBCS, for example).

To determine whether to use SIZE REDUCTION A or SIZE REDUCTION B, look at the macro invocation of TRXGN (top of the Rule Generator output for the ITR). If `PAD1=Y` is listed, use SIZE REDUCTION A; otherwise, use SIZE REDUCTION B.

For Segment Layout Rule (SLR)

The figure of 38 bytes (see Figure 91 on page 184) represents not only the savings for the SLR reduction, but also the savings for other SLR-related items in the SPA. These other items are as follows:

- 16 for segment (header) data
- Plus 6 for literals "*SLR* *" before each SLR in the SPA (you will see the SLR two-character rule identifier)
- Plus 8 for IMS segment name
- Plus 2 words of data per SLR

The constant of 27 bytes is made up of:

- Savings of 24 bytes for length of ITR header entry
- Plus 6 for literals in "** TR**", which is before the ITR in the SPA (you will see the literals IN)
- Plus 1 for extra byte of savings for ITRPATH "dummy" entry
- Plus 8 for length of "**STAB**" table header
- Plus 8-byte end of SLR absolute address table mark
- Minus 20 bytes for the "Reduced SPA" header

Appendix E. DB2 Space Allocation Calculations

This appendix lists the IMSADF II dynamic rules databases requirements for installation when you select DB2 as the access method. Use this appendix in conjunction with "DB2 Considerations" on page 9 to help you determine any additional space required for the applications you develop. The databases are described in Chapter 4 of *IMS Application Development Facility II Version 2 Release 2 Application Development Reference*; see that manual for more detail regarding the tables that comprise the databases. Space is reserved for the tables in IMSADF.JCLLIB(ADFTBLD), described in "ADFTBLD (Create Database/Tablespaces for DB2)" on page 54. If you must modify the allocated space based on calculations described in this appendix, change this member.

Installation-supplied jobs will group tables within a tablespace corresponding to their DL/I database equivalent. Tables are prefixed by a common letter.

S TSSIGN---Sign-on Profile database
A TSAUDIT--Audit database
M TSMSG----Message database

Figure 92 will help you allocate space for each tablespace.

TABLE NAME	DESCRIPTION	ROW LENGTH	# ADF ROWS	# YOUR ROWS	TOTAL ROWS	TOTAL K BYTES
S1	Project/Group	44	3			
S2	User Identifier	46	2			
S3	Profile Identifier	1216	2			
S4	Project Message	157	0			
A1	Operation Descriptor	38	55			
A2	Data Descriptor	48	43			
A3	Encode/Decode	100	0			
M1	Message Generation	78	148			
M2	Message Text	86	216			
M3	Help Error Messages	258	1			
M4	Message Routing	46	0			
M5	Help Text Screens	1652	9			
M6	Sec. Trans. Routing	150	0			
M7	User Mailbox	188	0			

Figure 92. DB2 Dynamic Rules Databases Calculations

The information listed in the figure is as follows:

- IMSADF II table name (TABLE NAME)
- A description of the information (DESCRIPTION)
- The length of each row (ROW LENGTH) in bytes
- The number of rows supplied at installation (# ADF ROWS)

Enter on the form the result of the following calculations:

- The number of rows you anticipate (# YOUR ROWS)
- Add # ADF ROWS and # YOUR ROWS together to arrive at (TOTAL ROWS)
- Compute (ROW LENGTH X TOTAL ROWS) / 1024
  to arrive at the number of kilobytes required for the table (TOTAL K BYTES)
- Sum the TOTAL K BYTES columns for all tables within the tablespace

Multiply the total by 2 to give an approximate amount of space required in that tablespace.

The multiplier 2 is chosen to allow for record overhead, free space within a page, unusable space on DASD due to blocking, unused space within data sets, and index space.

If you want to separate some of the tables into separate tablespaces, the exact formulae are explained in the DB2 manuals.

The space used by installation data is as follows:

- 5.2 KB--Sign-on Profile database
- 8.2 KB--Audit database
- 88.4 KB--Message database

Appendix F. Source Members Requiring Special Handling

This appendix lists the source members in IMSADF.ADFSLIB and IMSADF.ADFMAC requiring special handling.

Members of ADFSLIB

The following table lists the symbolic members of IMSADF.ADFSLIB that must be customized and the target library name. Tailor these members using the CUSTOMIZE option in the ISPF Installation Dialogs.

When maintenance is applied to a member of this library but you are not in the environment to which the member applies (see Chapter 5, “Running the Customized Installation Jobs”), you need not customize the member.

SYMBOLIC MEMBERS	CUSTOMIZED LIBRARY
ADFACBS	JCLLIB
ADFDATAS	JCLLIB
ADFDBDS	JCLLIB
ADFISPU	JCLLIB
ADFLINKS	JCLLIB
ADFOPTS	JCLLIB
ADFPSBS	JCLLIB
ADFRELDS	JCLLIB
ADFRL1	JCLLIB
ADFRL2	JCLLIB
ADFSTG1S	JCLLIB
ADFTBLB	JCLLIB
ADFTBLD	JCLLIB
ADFTBLDR	JCLLIB
ADFTBLF	JCLLIB
ADFTBLG	JCLLIB
ADFTBLT	JCLLIB
ADFTBTSO	JCLLIB
ADFTRTR	JCLLIB
ADFULNGS	JCLLIB
ADFUNLDS	JCLLIB
ADFUSERS	JCLLIB
ADFWORKI	JCLLIB
ADF2PAL	JCLLIB
BUILDT07	PSBSRC
BUILDT08	PSBSRC
BUILDT09	PSBSRC
BUILDT9B	PSBSRC
COPYPRCS	JCLLIB
DBDAUDS	DBDSRC
DBDMSGS	DBDSRC
DBDSIGNS	DBDSRC
DBDWORKS	DBDSRC
DBUPDATS	JCLLIB
DB2AUDIT	RULES.SOURCE

DB2BATCH	RULES.SOURCE
DB2BATSP	RULES.SOURCE
DB2BDDR	RULES.SOURCE
DB2DRUL	RULES.SOURCE
DB2MSG	RULES.SOURCE
DB2MSTR	RULES.SOURCE
DB2SIGNP	RULES.SOURCE
DB2SPRS	RULES.SOURCE
DB2STHR	RULES.SOURCE
DDEECI5	JCLLIB
DDEECI	JCLLIB
DDESAMPC	JCLLIB
JCLNLNKS	JCLLIB
MFSJOBS	JCLLIB
PROCALS	JCLLIB
PROCBS	JCLLIB
PROCGS	JCLLIB
PROCRDCS	JCLLIB
PSBBCTL	PSBSRC
PSBBCTP	PSBSRC
PSBBDDR	PSBSRC
PSBTCT	PSBSRC
PSBTOM	PSBSRC
PSBT99	PSBSRC
PSBV01	PSBSRC
PSBV05	PSBSRC
PSBV06	PSBSRC
PSBV07	PSBSRC
PSBWRKL	PSBSRC
RGLAUDIT	RULES.SOURCE
RGLBASE	RULES.SOURCE
RGLBATCH	RULES.SOURCE
RGLBATSP	RULES.SOURCE
RGLBDDR	RULES.SOURCE
RGLDB2S	RULES.SOURCE
RGLDRUL	RULES.SOURCE
RGLMSG	RULES.SOURCE
RGLSAMP	RULES.SOURCE
RGLSIGNP	RULES.SOURCE
SAMPTOR	PSBSRC
SAMPVCD	PSBSRC
SCREENS	JCLLIB
TRANSIN	JCLLIB
TRANSINF	JCLLIB
TRANSING	JCLLIB
TRANSINJ	JCLLIB
TRANSINK	JCLLIB
TRANSINP	JCLLIB
TRANSINS	JCLLIB
TRANSINW	JCLLIB
TRANSUP	JCLLIB
TRANSUPF	JCLLIB
TRANSUPG	JCLLIB
TRANSUPJ	JCLLIB

TRANSUPK	JCLLIB
TRANSUPP	JCLLIB
TRANSUPS	JCLLIB
TRANSUPW	JCLLIB
UCLDEL	JCLLIB

Members of ADFMAC (Special Processing Routines)

The following table lists the members of IMSADF.ADFMAC that are special processing routines, the component to which they apply, and the name of the source member in which the special processing routine is used.

See “Rules Documentation System” on page 108 for information on how to use this section.

SPR MEMBERS	COMPONENT	RULES.SOURCE
DFRFDR01	RDOC	RGLBDDR
DFRFDR02	RDOC	RGLBDDR
DFRFDR03	RDOC	RGLBDDR
DFRFDR04	RDOC	RGLBDDR
DFRFSR01	RDOC	RGLBDDR
DFRFSR02	RDOC	RGLBDDR
DFRFSR03	RDOC	RGLBDDR
DFRFSR04	RDOC	RGLBDDR
DFRFSR05	RDOC	RGLBDDR
DFRFSR06	RDOC	RGLBDDR
DFRFSR07	RDOC	RGLBDDR
MFC1UAM	BATCH DRIVER	RGLBATSP
MFC1UMM	BATCH DRIVER	RGLBATSP
SAMPUCD	SAMP	RGLSAMP
USERAUDT	SAMP	RGLSAMP

Appendix G. MFS Naming Conventions for Supplied Screens

This appendix shows the MFS naming standards for the screens generated by the SCREENS job when the ADFMFS macro is invoked.

```
&S. ==> ss = major system id (first two characters)
&T  ==>  t = MFS trailer (MFSTRLR)

PRIMARY OPTION MENU
  &S.TOM&T       BECOMES    ssTOMt       FORMAT NAME
  &S.MITOM&T     BECOMES    ssMITOMt     MID NAME
  &S.MOTOM&T     BECOMES    ssMOTOMt     MOD NAME

SECONDARY OPTION MENU
  &S.SOM&T       BECOMES    ssSOMt       FORMAT NAME
  &S.MISOM&T     BECOMES    ssMISOMt     MID NAME
  &S.MOSOM&T     BECOMES    ssMOSOMt     MOD NAME

SECONDARY KEY SELECTION
  &S.TKS&T       BECOMES    ssTKSt       FORMAT NAME
  &S.MITKS&T     BECOMES    ssMITKSt     MID NAME
  &S.MOTKS&T     BECOMES    ssMOTKSt     MOD NAME

PROJECT MESSAGE SENDING
  &S.TMS&T       BECOMES    ssTMSt       FORMAT NAME
  &S.MITMS&T     BECOMES    ssMITMSt     MID NAME
  &S.MOTMS&T     BECOMES    ssMOTMSt     MOD NAME

PROJECT MESSAGE DISPLAY
  &S.TPD&T       BECOMES    ssTPDt       FORMAT NAME
  &S.MITPD&T     BECOMES    ssMITPDt     MID NAME
  &S.MOTPD&T     BECOMES    ssMOTPDt     MOD NAME

USER MESSAGE SENDING
  &S.TUS&T       BECOMES    ssTUSt       FORMAT NAME
  &S.MITUS&T     BECOMES    ssMITUSt     MID NAME
  &S.MOTUS&T     BECOMES    ssMOTUSt     MOD NAME

USER MESSAGE DISPLAY
  &S.TUM&T       BECOMES    ssTUMt       FORMAT NAME
  &S.MITUM&T     BECOMES    ssMITUMt     MID NAME
  &S.MOTUM&T     BECOMES    ssMOTUMt     MOD NAME

ERROR SCREEN
  &S.TBB&T       BECOMES    ssTBBt       FORMAT NAME
  &S.MITBB&T     BECOMES    ssMITBBt     MID NAME
  &S.MOTBB&T     BECOMES    ssMOTBBt     MOD NAME

PRINTER SCREEN
  &S.T86&T       BECOMES    ssT86t       FORMAT NAME
  &S.MIT86&T     BECOMES    ssMIT86t     MID NAME
  &S.MOT86&T     BECOMES    ssMOT86t     MOD NAME
```

Figure 93. MFS Naming Conventions for Supplied Screens

Appendix H. Currency Requirements for Skipped Releases

If you installed IMSADF II Version 2 Release 1 (5665-348), this appendix does not apply to you.

While it is not recommended that you skip releases of IMSADF II, this appendix lists tasks you must do for IMSADF II Version 2 Release 2 to be properly installed when you currently have one of the following installed:

- 5796-PHX---IMSADF Release 1.3 (IUPX)
- 5668-937---IMSADF II Version 1

This appendix applies only to the IMS environment where DL/I is used to access the dynamic rules databases.

The tasks you must do **replace** installation job DBUPDATE. Two areas of concern exist, relating to dynamic rules databases currency and structure:

- Your Message database must be unloaded and reloaded because segments were added in Version 2 Release 1.
- Your dynamic rules databases do not have data supplied for each intervening release.

Structure Change of Dynamic Rules Databases

In IMSADF II Version 2 Release 1, two segments were added to the Message database for the HELP facility. If Release 2.1 is skipped, you must unload and reload the Message database.

You must also rerun your application PSBGENs after installation is complete. If a ? is entered in the OPTION field, IMSADF II accesses the new segments for HELP screens. Even if you do not have HELP text supporting existing applications (of course, you can add it at any time), the terminal operator will become accustomed to entering the HELP ? symbol. If your existing PSBs are not sensitive to the affected segments, you will receive a bad DL/I status code.

If the application PSB used the IMSADF II BUILDTxx macros to describe the PCBs for the IMSADF II databases, the PSBGEN will be correct if you direct the SYSLIB to the new IMSADF.PSBSRC when you generate the PSBs. Otherwise, you must change each PCB manually. For this reason, it is recommended that you use the BUILDTxx macros.

The Message database must be unloaded using the DBD from the old release and reloaded using the one you just created in job ADFDBD. These tasks must be done **before** you run job DBUPDATE.

Currency of Dynamic Rules Databases

Two members of IMSADF.JCLLIB supply data to the dynamic rules databases. These members have different functions.

- TRANSIN contains all data from inception of the product to the current release and is used by new IMSADF II users to initially **load** the dynamic rules databases.

- TRANSUP contains data added since the last release of the product and is used to **update** the dynamic rules databases of current IMSADF II installations.

TRANSIN is cumulative throughout releases, while the contents of TRANSUP are new with each IMSADF II release.

If you skip one or more releases of IMSADF II, you must incorporate all data added to the dynamic rules databases since your release.

It is a good idea to copy your dynamic rules databases and do the following sequence of events on the copy, leaving your original data sets intact. **First**, unload and reload your Message database, as described in the previous topic. Then do the following:

1. Run installation job DBUPDATE against your existing reorganized databases using TRANSIN as input.

   This adds all messages produced by IMSADF II since the release you installed. New data is inserted properly. You receive DL/I status codes for all supplied data that already exists. The return code is 20.

2. Make a separate copy of IMSADF.JCLLIB(TRANSIN) and do the following:

   a. Change all occurrences of `????B2` to `????B5`, where `????` is your four-character ADFID (TRANSIN was customized for your ADFID).

   b. Change all occurrences of `????B4` to `????B5`, where `????` is your four-character ADFID.

   This changes all inserts to update transactions.

3. Run DBUPDATE once more using your modified copy of TRANSIN as input. Since all data is now in update mode, the job completes with a return code of zero and verifies that all data supplied by IMSADF II is current.

Other Considerations

The following information is a brief summary of changes that occurred for Release 2.2. These changes should be reviewed for possible impact.

- Library names have changed, primarily to satisfy the SMP requirement that the lowest level name be the same as the ddname of the DLIB/SYSLIB. Additionally, all DLIBS that may appear in the same SMP procedure must have unique names; this is to avoid conflict with existing or any future products installable on the IMS SMP data sets.
- The object text library form for exits and special processing routines is no longer supported, so you must link edit yours after installation is complete. See the RGLIB parameter of DEFADF, described in Chapter 3, “Customization Macros (DEFADF and ADFMFS),” for more information.

Program Number: 5665-348

Printed in U.S.A.

SH20-6593-02

Spine information:

IBM

IMS Application Development Utility Installation Guide

Version 2 Release 2

SH20-6593-02