

Installing the FOCUS/IDMS Interface for MVS

The IDMS Read-Only Interface is available for the MVS batch, TSO interactive, and MSO environments. The Interface can be run under the IDMS Central Version or in Local Mode. It supports release 10.2, release 12, and release 14 of IDMS. A separate Interface is available to support the SQL option of CA-IDMS.

This Technical Memo describes how to install the IDMS Interface in the MVS and MSO environments for FOCUS Release 7.0.9, and contains the following sections:

- Section 1: The Distribution Tape Contents
- Section 2: Installation Process
- Section 3: User Written Exits
- Appendix A: Accessing IDMS Databases in Local Mode and Central Version

Related publications which you may find useful include:

- IDMS/R Interface Users Manual (DN1000049.0888)
- FOCUS for IBM Mainframe MVS/TSO Installation Guide, Release 7.0 (DN1000994.1097)
- FOCUS for IBM Mainframe Multi-Session Option Installation and Technical Reference Guide, Release 7.0 (DN1000966.1095)

SECTION 1: THE DISTRIBUTION TAPE CONTENTS

The IDMS Read-Only Interface is distributed on the same tape/cartridge as the base FOCUS product.

The table below shows the file number and data set attributes for the files needed to use the FOCUS/IDMS interface:

| FILE # | DATASET NAME | TYPE | LRECL | BLKSIZE | RECFM |
|--------|--------------|------|-------|---------|-------|
| 9 | IDMS.LOAD | PDS | 13030 | 13030 | U |
| 10 | IDMS.DATA | PDS | 80 | 1600 | FB |

These files are partitioned data sets (PDS) in IEBCOPY dump format.

SECTION 2: INSTALLATION PROCESS

Install the IDMS Interface by following the steps listed below:

1. Customize the IEBCOPY JCL
2. Prepare the run-time libraries
3. Customize the FOCUS CLIST
4. Install the AUTOIDMS facility

SECTION 2: INSTALLATION PROCESS

2.1 Customizing the IEBCOPY JCL

There are two datasets on the distribution tape which were created using the IEBCOPY utility. Create an IEBCOPY procedure like the one listed below and submit it for execution.

```
//COPYPDS EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//INDD1 DD DSN=IDMS.DATA,DISP=OLD,
// UNIT=unit,VOL=SER=volser,LABEL=(10,SL)
//INDD2 DD DSN=IDMS.LOAD,DISP=OLD,
// UNIT=unit,VOL=SER=volser,LABEL=(9,SL),
//OUTDD1 DD DSN=prefix.IDMS.DATA,DISP=(,CATLG,DELETE),
// UNIT=SYSDA,SPACE=(TRK,(50,10,50))
//OUTDD2 DD DSN=prefix.IDMS.LOAD,DISP=(,CATLG,DELETE),
// UNIT=SYSDA,SPACE=(CYL,(2,1,50),RLSE)
//SYSIN DD *
COPY INDD=INDD1,OUTDD=OUTDD1
COPY INDD=INDD2,OUTDD=OUTDD2
/*
```

where:

unit

tape/cartridge

volser

tape/cartridge serial number

prefix.IDMS.LOAD

dataset name of load library

prefix.IDMS.DATA

dataset name of source library

Upon successful completion there are two new PDS members cataloged on your system. The dataset names for these members will depend on what you specified on your IEBCOPY OUTDD1 and OUTDD2 ddcards.

2.2 Preparing the Run-time Libraries

1. Using your site criteria, allocate a partitioned dataset (PDS) for your Access File Descriptions as ddname FOCIDMS. This library (or libraries) can be a mirror image of ddname MASTER and should have the same DCB parameters. Refer to the FOCUS for IBM Mainframe MVS/TSO Installation Guide, Release 7.0 for assistance in selecting DCB parameters. Two suggested names for this dataset are:

prefix.ACCESS.DATA or

prefix.FOCIDMS.DATA

2. The FOCUS description of an IDMS database is called a Master File Description. It is stored as a member of a PDS which is allocated to ddname MASTER in the FOCUS CLIST/JCL. Refer to the FOCUS for IBM Mainframe MVS/TSO Installation Guide, Release 7.0 for assistance allocating this PDS.

SECTION 2: INSTALLATION PROCESS

3. A stored FOCUS procedure is called a FOCEXEC. A FOCEXEC is stored as a member of a PDS which is allocated to ddname FOCEXEC in the FOCUS CLIST/JCL. Refer to the FOCUS for IBM Mainframe MVS/TSO Installation Guide, Release 7.0 for assistance allocating this PDS.

2.3 Customizing the FOCUS CLIST

The CLIST used to execute the IDMS/FOCUS Interface must be modified to include the Access File Description PDS and the PDS containing the FOCUS/IDMS Interface load modules. The 'prefix.IDMS.LOAD' dataset, defined in Section 2.1, includes the FOCUS/IDMS load modules that must be concatenated to the USERLIB, FOCLIB or STEPLIB ddnames. It is important to note that if multiple copies of the IDMS Interface exist, the load module will be loaded from USERLIB first, followed by FOCLIB then STEPLIB. For batch jobs, however, allocate all load libraries to the ddname STEPLIB.

The 'prefix.IDMS.DATA' dataset, defined in Section 2.1, contains a member named FOCIDMS which contains the FOCUS/IDMS error messages. This dataset must be concatenated to the ERRORS ddname in the FOCUS CLIST. Failure to concatenate this dataset to the ERRORS ddname results in the following message when using the Interface:

`ERROR TEXT MISSING`

The CLIST or JCL must contain these allocations:

1. The PDS containing the FOCUS/IDMS Access File Descriptions.
2. The PDS containing the FOCUS/IDMS Master File Descriptions.
3. The PDS containing the FOCUS procedures (FOCEXECs).
4. The PDS containing the FOCUS ERROR messages.
5. The PDS containing the FOCUS/IDMS Interface load modules.
6. The IDMS run-time load modules (refer to Appendix A).

The following chart identifies the allocations needed to execute a FOCUS procedure (FOCEXEC) against an IDMS database:

| PDS CONTAINING MEMBERS: | CONCATENATE TO DDNAME: |
|--------------------------------|-------------------------------|
| Access File Descriptions | FOCIDMS |
| Master File Descriptions | MASTER |
| FOCUS Procedures | FOCEXEC |
| Error Messages | ERRORS |
| Interface Load Module | * USERLIB, FOCLIB or STEPLIB |
| FOCUS Load Modules | |
| IDMS Run-Time Load Modules | Depends on local or CV access |

* For batch jobs, allocate all load modules to the ddname STEPLIB.

SECTION 2: INSTALLATION PROCESS

2.4 Linking the FOCUS/IDMS Interface to Run Below the Line (IDMS Release 10.2 Only)

If you are running CA-IDMS release 10.2, you must re-link the IDMSR module to run below the line. Following is a sample of the JCL needed to re-link:

```
//jobcard goes here
//LKED EXEC PGM=IEWL,PARM='LET,LIST,NCAL,XREF'
//OLDMOD DD DSN=prefix.IBI.IDMS.LOAD,DISP=SHR
//SYSLMOD DD DSN=prefix.IBI.IDMS.LOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD UNIT=SYSDA,SPACE=(100,(500,500))
//SYSLIN DD *
        MODE AMODE(24),RMODE(ANY)
        ENTRY IDMSR
        INCLUDE OLDMOD(IDMSR)
        NAME IDMSR(R)
/*
```

where:

`prefix.IDMS.LOAD`

Is the FOCUS/IDMS Interface load library supplied by Information Builders.

2.5 Installing the AUTOIDMS facility

The 'prefix.IDMS.DATA' dataset contains four members that must be renamed and copied into the appropriate libraries according to the chart below:

| MEMBER NAME: | RENAME TO: | COPY TO: |
|---------------------------|-------------------|---------------------|
| IDMSIDDM or IDMSI10M * | IDMSIDD | prefix.MASTER.DATA |
| AUTOIDMM | AUTOIDMS | prefix.MASTER.DATA |
| IDMSIDDA | IDMSIDD | prefix.ACCESS.DATA |
| AUTOIDMS | AUTOIDMS | prefix.FOCEXEC.DATA |

* Member IDMSI10M is the Master File that describes the CA-IDMS/DB 10.2 dictionary. When renamed to IDMSIDD, AUTOIDMS correctly describes your release 10.2 subschemas.

SECTION 2: INSTALLATION PROCESS

The following JCL is located in member GENFAUTO in the 'prefix.IDMS.DATA' dataset. This JCL unloads and renames the members needed for the AUTOIDMS facility. Be sure to remove all other references to the old AUTOIDMS facility before you execute the IEBCOPY job.

```
//COPYPDS EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//INDD1 DD DSN=prefix.IDMS.DATA,DISP=SHR
//OUTDD1 DD DSN=prefix.MASTER.DATA,DISP=SHR
//OUTDD2 DD DSN=prefix.ACCESS.DATA,DISP=SHR
//OUTDD3 DD DSN=prefix.FOCEXEC.DATA,DISP=SHR
//SYSIN DD *
COPY INDD1,OUTDD=OUTDD1
SELECT MEMBER=((IDMSIDDM,IDMSIDD))
COPY INDD1,OUTDD=OUTDD1
SELECT MEMBER=((AUTOIDMM,AUTOIDMS))
COPY INDD=INDD1,OUTDD=OUTDD2
SELECT MEMBER=((IDMSIDDA,IDMSIDD))
COPY INDD=INDD1,OUTDD=OUTDD3
SELECT MEMBER=((AUTOIDMS,,R))
```

where:

| | |
|---------------------|--|
| prefix.IDMS.DATA | Unloaded PDS from the install tape |
| prefix.MASTER.DATA | PDS containing the FOCUS Master File Descriptions |
| prefix.ACCESS.DATA | PDS containing the FOCUS/IDMS Access File Descriptions |
| prefix.FOCEXEC.DATA | PDS containing the FOCUS procedures |

2.6 AUTOIDMS and Secondary Dictionaries

By default, AUTOIDMS searches for the subschema descriptions in the primary dictionary. However, subschemas frequently reside in dictionaries other than the primary dictionary. These are called secondary dictionaries. If a subschema that is being described by AUTOIDMS is located in a secondary dictionary, a DBNAME= and a DICTNAME= parameter is needed in member IDMSIDD of the AUTOIDMS Access File Description. Section 2.6 describes how to set up a secondary dictionary environment.

2.7 Determining Which IDD Dictionaries Will Be Accessed

If there is only one primary dictionary used to generate descriptions, skip to Section 3. This step is to setup AUTOIDMS to access several IDD secondary dictionaries (these are the dictionaries where the subschema resides).

1. Determine which IDD dictionaries are accessed by AUTOIDMS. For each dictionary, select a one character identifier. This value is used to uniquely identify the dictionary for reporting purposes and is displayed on the Main Menu of AUTOIDMS. The identifier may be the letter A through Z, the number 0 through 9, or blank. For example, if you have a test and production IDD dictionary, you might select T and P as the identifiers.

SECTION 2: INSTALLATION PROCESS

- Copy and rename the following files, once for each suffix selected in the previous step (where n represents the identifier):

| | |
|-----------------------------|------------------------------|
| From: | To: |
| prefix.MASTER.DATA(IDMSIDD) | prefix.MASTER.DATA(IDMSIDDn) |
| prefix.ACCESS.DATA(IDMSIDD) | prefix.ACCESS.DATA(IDMSIDDn) |

- Edit the ACCESS.DATA(IDMSIDDn) to include the appropriate DBNAME and DICTNAME on the first line. For example

```
SSHEMA=IDMSNWKA,RELEASE=12,DBNAME=SYSDIRL,DICTNAME=SYSDIRL,$
```

- Edit prefix.FOCEXEC.DATA(AUTOIDMS). Locate the line that reads:

```
-DEFAULT &DICT_FILES=' '
```

and change the value to include each of the identifiers selected in Step 1 of this section, and only those values. The first character is used as the default dictionary on the AUTOIDMS Main Menu. For example, if you chose P and T as your identifiers and you wish to have the production dictionary as the default, the line would read:

```
-DEFAULT &DICT_FILES='PT '.
```

Note that the characters must be entered in upper case.

- Repeat Step 2.4 above for each of the identifiers that were selected in Step 1 (from Section 2.6.).

SECTION 3: USER WRITTEN EXITS

The FOCUS/IDMS Interface has the ability to execute user written programs via the ZBIND and ZREADY exit facilities.

These programs can be written in any IDMS supported language and must be link edited to the FOCUS/IDMS load module IDMSR.

The user written programs initiated by the ZBIND and ZREADY exits must be link edited as AMODE(24), RMODE(24).

The ZBIND and ZREADY exits can call any user-specified program. The FOCUS/IDMS Interface passes back to FOCUS a 4 byte status code. An error condition occurs when this status code is not equal to zero.

3.1 IDMS ZBIND Exit

The program that is link edited with the FOCUS/IDMSR module with the name ZBIND is called via standard IBM calling conventions. When a user issues a FOCUS TABLE request against an IDMS database, the FOCUS/IDMS Interface calls the ZBIND exit before issuing the IDMS BIND RUN UNIT command.

SECTION 2: INSTALLATION PROCESS

A sample COBOL program, located on the tape as member ZBINDPGM in the 'prefix.IDMS.DATA' PDS, can be used to check if a user has authorization to access the subschema that is associated with the FOCUS request. There are six parameters passed to the program from the FOCUS/IDMS Interface:

```
USER          PIC X(08)
SUBSCHEMA     PIC X(08)
DBNAME        PIC X(08)
NODE          PIC X(08)
DICTNAME      PIC X(08)
DICTNODE      PIC X(08)
STATUS        PIC X(04)
```

On return of a non-zero status code, the user receives the following error message:

```
(FOC4405) BIND RUN-UNIT DENIED BY USER EXIT ZBIND FOR SUBSCHEMA: subschema
```

The following is a listing of the ZBINDPGM program provided on the tape:

```
*RETRIEVAL
*NO-ACTIVITY-LOG
  IDENTIFICATION DIVISION.
  PROGRAM-ID. ZBIND.
*****
* PROGRAM: ZBIND
* PURPOSE: CHECK TO SEE IF A USER ISSUING A FOCUS REQUEST
* HAS ACCESS TO THE SUBSCHEMA REQUESTED.
* THIS PROGRAM MAY BE USED FREELY BY THOSE
* WHO NEED AN EXAMPLE TO CODE THEIR OWN EXIT.
*****
  DATE-WRITTEN. AUG 1992.
  DATE-COMPILED. DATE COMPILED
  ENVIRONMENT DIVISION.
  CONFIGURATION SECTION.
  SOURCE-COMPUTER. IBM-370.
  SPECIAL-NAMES.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
  IDMS-CONTROL SECTION.
  PROTOCOL. MODE IS BATCH DEBUG IDMS-RECORDS MANUAL.
  SCHEMA SECTION.
  DB IDMSNWK WITHIN IDMSNTWK.
  DATA DIVISION.
  EJECT
  FILE SECTION.
  EJECT
  WORKING-STORAGE SECTION.
  01 WS-START.
     02 FILLER PIC X(33) VALUE
        'ZBIND WORKING STG STARTS HERE==>'.
  01 COPY IDMS SUBSCHEMA-CONTROL.
  01 COPY IDMS RECORD USER-047.
  01 COPY IDMS RECORD ACCESS-045.
  01 COPY IDMS RECORD SS-026.
  LINKAGE SECTION.
```

SECTION 2: INSTALLATION PROCESS

```
01 PUSER          PIC X(8).
01 PSS            PIC X(8).
01 PDBNAME        PIC X(8).
01 PNODE          PIC X(8).
01 PDICT          PIC X(8).
01 PDICTNODE      PIC X(8).
01 PSTAT          PIC X(4).
*****
*
* RETURN CODE - 0000 - USER HAS ACCESS TO SUBSCHEMA
*              - 0016 - USER NOT IN DICTIONARY
*              - 0032 - USER DOES NOT HAVE ACCESS TO SUBSCHEMA
*              - 9999 - PASSED PARMS USER AND/OR SS INVALID
*              - NNNN - ERROR DURING IDMS-STATS PROCESSING.
*                  WHERE NNNN = ERROR-STATUS
*
*****
PROCEDURE DIVISION USING PUSER,
                    PSS,
                    PDBNAME,
                    PNODE,
                    PDICT,
                    PDICTNODE,
                    PSTAT.

0000-MAINLINE.
*** DISPLAY ' ' PUSER ' '
***          ' ' PSS  ' '.
*** RESET ERROR-STATUS TO ENSURE REENTRANCY
    MOVE '1400' TO ERROR-STATUS.
    MOVE 'ZBIND' TO PROGRAM-NAME.
*** CHECK FOR INVALID PASSED PARMS
    IF PUSER = SPACE OR
       PSS = SPACE
       MOVE '9999' TO PSTAT
       GOBACK.
*** CHECK FOR PRESENCE OF DBNAME PARM - DETERMINE WHICH BIND
    NOTE: THIS PROGRAM ASSUMES THE DBNAME PASSED IS THE DBNAME
          WHICH MAPS THE IDMSNWKA SUBSCHEMA TO THE REQUESTED
          DICTIONARY.
          IT DOES ==> NOT <== LOOK AT THE DICTNAME PARAMETER
          PASSED
          TO THE EXIT.
    IF PDBNAME = SPACE
       BIND RUN-UNIT
    ELSE
       BIND RUN-UNIT DBNAME PDBNAME.
    IF ANY-ERROR-STATUS
       MOVE ERROR-STATUS TO PSTAT
       GOBACK.
*** BIND AND READY
    BIND USER-047.
    IF ANY-ERROR-STATUS
       MOVE ERROR-STATUS TO PSTAT
```

SECTION 2: INSTALLATION PROCESS

```
GO TO 9999-WRAP-UP.
BIND ACCESS-045.
IF ANY-ERROR-STATUS
  MOVE ERROR-STATUS TO PSTAT
  GO TO 9999-WRAP-UP.
BIND SS-026.
IF ANY-ERROR-STATUS
  MOVE ERROR-STATUS TO PSTAT
  GO TO 9999-WRAP-UP.
READY USAGE-MODE IS RETRIEVAL.
IF ANY-ERROR-STATUS
  MOVE ERROR-STATUS TO PSTAT
  GO TO 9999-WRAP-UP.
*** SEE IF USER IS IN THE DICTIONARY
MOVE PUSER TO USER-NAME-047.
FIND CALC USER-047.
IF DB-REC-NOT-FOUND
  MOVE '0016' TO PSTAT
  GO TO 9999-WRAP-UP.
IF ANY-ERROR-STATUS
  MOVE ERROR-STATUS TO PSTAT
  GO TO 9999-WRAP-UP.
*** LOOP THRU USER-ACCESS-SS RECS TO SEE IF RELATIONSHIP EXISTS
1200-NEXT-ACCESS.
  FIND NEXT ACCESS-045 WITHIN USER-ACCESS.
  IF DB-END-OF-SET
    MOVE '0032' TO PSTAT
    GO TO 9999-WRAP-UP.
  IF ANY-ERROR-STATUS
    MOVE ERROR-STATUS TO PSTAT
    GO TO 9999-WRAP-UP.
  IF NOT SS-ACCESS MEMBER
    GO TO 1200-NEXT-ACCESS.
  OBTAIN OWNER WITHIN SS-ACCESS.
  IF ANY-ERROR-STATUS
    MOVE ERROR-STATUS TO PSTAT
    GO TO 9999-WRAP-UP.
  IF SS-NAM-026 = PSS
    MOVE '0000' TO PSTAT
    GO TO 9999-WRAP-UP.
  GO TO 1200-NEXT-ACCESS.
9999-WRAP-UP.
FINISH.
GOBACK.
EJECT
*COPY IDMS IDMS-STATUS.
*IDMS-ABORT SECTION.
*I-A-EXIT. EXIT.
```

SECTION 2: INSTALLATION PROCESS

The following JCL, member ZBIND in the 'prefix.IDMS.DATA' PDS can be used to link edit the ZBIND user written program.

```
//LINK      EXEC PGM=IEWL,
//          PARM='LET,NCAL,SIZE=(1024k),LIST'
//OBJLIB   DD DSN=prefix.ZBIND.OBJLIB,DISP=SHR
//LOAD     DD DSN=prefix.IDMS.LOAD,DISP=SHR
//SYSLMOD  DD DSN=prefix.OUTPUT.LOADLIB,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(10,1))
//SYSPRINT DD SYSOUT=*
//SYSLIN   DD *
INCLUDE OBJLIB(ZBIND)
INCLUDE LOAD(IDMSR)
ENTRY IDMSR
NAME IDMSR(R)
/*
//
```

where:

| | |
|-----------------------|--|
| prefix.ZBIND.OBJLIB | PDS containing ZBIND object code |
| prefix.IDMS.LOAD | PDS containing the IDMSR load module |
| prefix.OUTPUT.LOADLIB | PDS to contain the new IDMSR load module |

3.2 IDMS ZREADY Exit

The program that is link edited with the FOCUS/IDMSR module with the name ZREADY is called via standard IBM calling conventions.

When a user issues a FOCUS TABLE request against an IDMS database, the FOCUS/IDMS Interface calls the ZREADY exit before issuing an IDMS READY command.

An example of a useful user written program is one that resolves conflicts between FOCUS users who have access to IDMS databases through the FOCUS/IDMS Interface and maintenance jobs which require exclusive control of their IDMS areas. Often such jobs cannot be initiated because they are waiting for FOCUS users to finish processing. Even though a single FOCUS request may only take a short time, new requests might be started so often that required areas may not become free for a long time. This ZREADY exit provides a way to inhibit new FOCUS requests, and also to force cancellation of active requests if they do not finish within a prescribed time.

The FOCUS/IDMS Interface calls the ZREADY exit for each FOCUS TABLE request against an IDMS database, once for each area containing fields referenced in the FOCUS request. This gives the user exit an opportunity to determine whether or not the FOCUS/IDMS Interface should READY the area in question, and if so, to retain this fact for possible subsequent action. If ZREADY indicates that the READY is not to be issued, the Interface informs the FOCUS user and terminates that particular request without retrieving any data, even if the areas were already readied. After this it ends its contact with IDMS by issuing a FINISH command. The FOCUS session itself is not ended; the user is free to try to access other IDMS areas.

SECTION 2: INSTALLATION PROCESS

Call ZREADY with the following parameters:

```
SUBSCHEMA   PIC X(08)
AREANAME    PIC X(32)
DBNAME      PIC X(08)
NODE        PIC X(08)
DICTNAME    PIC X(08)
DICTNODE    PIC X(08)
STATUS      PIC X(04)
```

On return of a non-zero STATUS code, the user receives the following error message:

```
(FOC4382) READY INHIBITED BY USER EXIT ZREADY FOR AREA: areaname
```

The following JCL, member ZREADY in the 'prefix.IDMS.DATA' PDS can be used to link edit the ZREADY user written program.

```
//LINK      EXEC PGM=IEWL,
//          PARM='LET,NCAL,SIZE=(1024k),LIST'
//OBJLIB    DD DSN=prefix.ZREADY.OBJLIB,DISP=SHR
//LOAD      DD DSN=prefix.IDMS.LOAD,DISP=SHR
//SYSLMOD   DD DSN=prefix.OUTPUT.LOADLIB,DISP=SHR
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(10,1))
//SYSPRINT  DD SYSOUT=*
//SYSLIN    DD *
INCLUDE OBJLIB(ZREADY)
INCLUDE LOAD(IDMSR)
ENTRY IDMSR
NAME IDMSR(R)
/*
//
```

where:

| | |
|-----------------------|--|
| prefix.ZREADY.OBJLIB | PDS containing ZREADY object code |
| prefix.IDMS.LOAD | PDS containing the IDMSR load module |
| prefix.OUTPUT.LOADLIB | PDS to contain the new IDMSR load module |

APPENDIX A: ACCESSING IDMS DBMS IN LOCAL MODE AND CENTRAL VERSION

Instructions for accessing IDMS databases differ for Local Mode access and Central Version (CV) access.

Central Version access:

The ddname SYSCTL must be allocated to the SYSCTL dataset corresponding to the Central Version desired. No allocation of IDMS data files or subschema/DMCL load modules are needed when running CV.

SECTION 2: INSTALLATION PROCESS

The IDMS functions will take place in the IDMS CV address space. The subschema load modules are located and retrieved in the following order:

1. CV primary load area (DDLDCLOD area)
2. CDMSLIB ddname from the IDMS Central Version job
3. STEPLIB ddname from the IDMS Central Version job

The CV's global DMCL is used but may be overridden by the DCML assigned in the SYSIDMS ddname (IDMS rel 12.x).

Note: When accessing data from secondary dictionaries the primary load area is not searched to retrieve the subschema/DMCL modules.

Local Mode access:

The user must allocate all IDMS database files. These files must be allocated to their respective ddnames that are assigned in the CA-IDMS/DB Schema.

All journal file allocations must be made available along with the default local mode journal, SYSJRNL, assigned to DD DUMMY.

When running the interface in a batch job or from an MSO server, the load modules must be allocated to the ddname STEPLIB.

In some cases, the libraries containing the subschema, DMCL, and IDMSINTB load modules may not be authorized. Because of some restrictions of STEPLIB and authorization, the CDMSLIB ddname could be used in a local mode job. With CDMSLIB allocated, IDMS will search the CDMSLIB first, before STEPLIB, to obtain all CA/IDMS-DB specific load modules.

When executing AUTOIDMS in local mode and not using to the primary dictionary, the DBNAME=, DICTNAME= parameters specified in the IDMSIDD Access File searches the specified secondary dictionary. The CA/IDMS-DB database name table (IDMSDBTB) Load Module member is used. The CA/IDMS-DB load library containing this member must be allocated to the ddname STEPLIB or ISPLLIB in the CLIST, or to STEPLIB in a batch or MSO environment. The secondary dictionary dataset must also be allocated to the appropriate ddname assigned in the CA-IDMS/DB schema.

For either mode:

The CA-IDMS load modules, IDMS and IDMSINTB must be made available at run-time allocated to the ddname STEPLIB.

The member names of the FOCUS Master File Description and Access File Descriptions to read the subschema must be identically named and made available at run time.

If you are running IDMS release 12.0, SYSIDMS can be allocated to identify the DCML for both CV and LOCAL mode.