BMC Shines a Light on the IBM® IMS™ Catalog

Catalog Manager for IMS™

By Robert Blackhall and Duane Wente
# Table of Contents

1. EXECUTIVE SUMMARY

2. INTRODUCTION
   Challenges Prior to Using the IMS Catalog

3. TOP SIX REASONS TO USE THE IMS CATALOG

4. TOP SIX RISKS IN USING THE IMS CATALOG

   ENABLING THE IMS CATALOG
   - Components of IMS Catalog Deployment
   - Process for Enabling the IMS Catalog
   - Enabling the IMS Catalog for Batch Processing
   - Enabling IMS-Managed ACBs

6. CATALOG CONSIDERATIONS
   - Shared IMS Catalog Versus Non-Shared IMS Catalog
   - Sharing the IMS Directory Between IMS Systems on Different LPARS
   - Deploying IMS-Managed ACBs
   - Catalog Out of Sync
   - Directory Out of Sync
   - Catalog Purge Utility
   - Migrating from IMS 14 to IMS 15
   - Catalog API

9. ADDRESSING THE IMS CATALOG CHALLENGES WITH A BMC SOLUTION
   - Catalog Manager for IMS
   - Catalog Viewing
   - Catalog Reports
   - JCL Generation
   - IMS Command Facility
   - Integration with the DBA Toolkit
   - Future Direction for Catalog Manager for IMS
IBM® IMS™ catalog (IMS catalog) is a game-changer in the IMS space, moving the IMS platform forward to ensure long-term IMS viability. As you consider the IMS Catalog, you must keep in mind that implementing the catalog and directory is not an easy exercise. It requires that you test, architect, and rethink how you manage IMS. As a result, this change is creating a great deal of concern.

This white paper provides information on the history of the IMS catalog and the key reasons to use the catalog along with the top risks you may encounter as you start to implement the catalog. Since most sites are just starting to investigate the catalog, there is detailed information on the steps you need to follow to set up the IMS catalog and get it operational. The last section shows the BMC offering that assists in the implementation and ongoing management of the IMS catalog environment with the ability to view catalog information, produces reports comparing the catalog to your control block libraries, and generates jobs to update the IMS catalog.
INTRODUCTION

The IMS catalog has been available since 2012 with IBM IMS 12 for Oracle Java SQL access using metadata. It provides a trusted online source for two types of metadata information, IMS database and application metadata. IMS 13 added database versioning to the catalog to allow changes to a database without changing an application. IMS 14 added IMS managed application control blocks (ACBs) and the directory.

Currently, there are installations using the IMS catalog for Java or database versioning, while others are also using IMS managed ACBs. Eventually, all installations will need to use both the IMS catalog and IMS managed ACBs.

The IBM announcement for IMS 15 has a statement of future direction that IMS will require IMS-managed ACBs for the IMS online environment. This has caused many IMS customers to start investigating the catalog to determine how to use it in their environments and the impact it will have on their current processes and procedures.

Initial responses from customers have been mixed:
• For some, this seems like the next step as the IMS catalog will provide added functionality that will allow users to manage IMS more effectively with a catalog, SQL access of IMS data, and DDL to define IMS databases.
• For others, especially with IMS resources being in high demand, this seems like a lot of work to successfully implement the catalog in their environments.

Challenges Prior to Using the IMS Catalog
• Primary function of the IMS catalog is to enable scalable and flexible IMS open database solutions.
• Prior to the IMS catalog, applications written in Java needed to maintain local IMS database and application metadata.
• Keeping the distributed and mainframe versions of the same information in-sync proved to be error-prone and resulted in distributed application outages due to metadata inconsistencies when IMS control block changes were implemented.

The following figures illustrate the difference in the information before and after using the IMS catalog.

Before the IMS catalog (different versions of data in the mainframe and distributed environment):
After the IMS catalog (one single source of truth):

As you consider the IMS Catalog, you must keep in mind that implementing the catalog and directory is not an easy exercise. It requires that you test, architect, and rethink how you manage IMS. As a result, this change is creating a great deal of concern.

**TOP SIX REASONS TO USE THE IMS CATALOG**

These are the six compelling reasons why you must use the IMS catalog:

1. IBM will require you to use the catalog with IMS. IMS managed ACBs is the strategic direction for IMS 15 and any subsequent continuous-delivery functionality. If you do not already use the IMS catalog, you must enable the catalog when you enable IMS management of ACBs.

   As stated in the IMS 15 announcement (IBM United States Software Announcement 217-398) dated October 3, 2017, “IBM intends to require IMS management of ACBs in the future. IMS and the IMS catalog must be set up to support ACB management. IMS provides a utility for this. After the requirement for IMS-managed ACBs is in place, IBM also intends to remove the generation processes for PSBLIB, DBDLIB, and ACBLIB. At that time, the IMS catalog, SQL, and DDL become the interface to IMS database management.”

2. As the role of IMS application databases expands and complements the multi-tiered architecture, mission-critical processing that requires unparalleled performance will continue to be served by the IMS hierarchical model. Analytics and business intelligence processes are best served by the relational model. Many large companies are beginning to use both models. Integration capabilities enabled via the IMS catalog position IMS at the center of this multi-platform environment to ensure consistency in both cases.

3. The IMS catalog and managed ACBs will reduce the number of GENs and provide a central repository for all IMS data structures. Reduction in the operational cost and usage of single source of truth for DBD and PSB metadata enables wider acceptance for IMS back-end support for distributed applications. This will reduce the risk associated with migration off IMS of applications better suited for high-performance transaction processing, while at the same time enabling analytics and business intelligence requirements to be met.

4. Catalog metadata will enhance the understanding of data housed in IMS and used by all applications that need to use this data. Data consistency and governance across the organization, impact analysis, and development productivity will also improve.
5. IMS catalog lays the groundwork for a comprehensive change management platform that allows not only database versioning but also a comprehensive repository for IMS database objects and applications. Comparing between the IMS catalog metadata entries from different environments provides visibility into unintended IMS object changes and usage.

6. The IMS catalog is accessible from distributed platforms. This eliminates the need to recode and maintain local IMS objects. It also eliminates a source of inconsistencies and simplifies access to IMS objects on the mainframe.

**TOP SIX RISKS IN USING THE IMS CATALOG**

These are the risks you need to manage if you choose to use the IMS catalog:

1. IMS maintenance procedures might require changes to make them stable. The addition of the Catalog Populate utility to the ACBGEN process will affect the procedures that installations use to make changes to ACBLIB. Not using this utility after ACBGEN will cause catalog out-of-sync conditions with ACBLIB.

2. Migrating to a new IMS version has always required an ACBGEN BUILD PSB=ALL to generate an ACBLIB for the new IMS version that is not compatible with earlier versions. When using a catalog shared between IMS versions, ACBGEN will result in the Catalog Populate utility inserting a new instance of both PSB and DBD into the catalog, increasing the size of the catalog.

3. PSBLIB and DBDLIB can be out of sync with ACBLIB for the batch environment. ACBLIB can be out of sync with the active ACB library for the IMS online environment. With IMS managed ACBs, the directory must be in sync with ACBLIB and the catalog.

4. There are also potential out-of-sync possibilities when rolling back a change by using the Catalog Purge utility.

5. Catalog size will increase significantly with the addition of fields to segments used by the application but not previously defined in the DBD. You must use proactive database management for space and performance for the catalog database.

6. There is a risk in not utilizing DBRC. Using DBRC to define the catalog HALDB partitions is recommended as DBRC provides advantages in data sharing and backup and recovery.

**ENABLING THE IMS CATALOG**

The sections that follow describe the components and the process for enabling the IMS catalog.

**Components of IMS Catalog Deployment**

- IMS catalog database—an IMS OSAM PHIDAM database that contains information about IMS databases and application programs.
- IMS-managed ACBs—management of IMS ACB (database and application) control blocks. This management uses the IMS catalog as the source feed to populate a new IMS catalog component called the IMS directory.
- IMS directory—a collection of PDSEs that will contain members in a format similar (but not identical to) the ACB members in an ACBLIB library. IMS manages the name and number of IMS directory data sets. The IMS directory data sets are an extension of the IMS catalog, and as such, will belong only to that single IMS catalog occurrence.

**Process for Enabling the IMS Catalog**

1. Copy the provided IMS catalog DBDs and PSBs to your DBDLIB and PSBLIB as follows:
   a. DBDs—DFSCD000, DFSCX000
   b. PSBs—DFSCPL00, DFSCP000, DFSCP001, DFSCP002, DFSCP003

2. Perform ACBGEN to generate the IMS catalog ACB members.

3. Update your DFSDFxxx PROCLIB member to contain a <SECTION=CATALOG> as follows:

   - CATALOG=YES /* enables IMS catalog */
   - ALIAS=DFSC /* use DFSC if no alias name value */
   - DATACLASS=,MGMTCLAS=,STORCLAS= /* SMS data set allocation settings */
   - SMSVOLCT=n /* Number of volumes */
   - SPACEALLOCO=(PRIMARY=nn1,SECONDARY=nn2) /* IMS catalog allocation parameters */
   - RETENTION(INSTANCES=nn1,DAYS=nn2) /* Define instance purge criteria */
For example:

```
<CSECTION=CATALOG>
CATALOG=Y
ALIAS=DFSC
STORCLAS=DEVSMS
SMSVOLCT=3
SPACEALLOC=(PRIMARY=500,SECONDARY=50)
RETENTION INSTANCES=5,DAYS=180
ACBMGMT=CATALOG
</SECTION>
```

4. Define the IMS catalog HALDB master and partition definition to DBRC. If an IMS catalog alias is specified, each alias catalog database must be defined to DBRC.

For example:

```
INIT.DB  DBD(DFSCD000) TYPHALDB SHARELVL(3)
INIT.PART DBD(DFSCD000) PART(DFSCD01) -
            DSNPREFIX(DBU.QA.MX01.DFSCD000) -
            KEYSRNG(X'FFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
            FFFFFFFFFFFFFFFFFFFFFFFFFF-
b. Assuming the estimated size is appropriate for your system, run the IMS Catalog Populate utility with DFSCPL00 to load the catalog. If the size is not large enough, return to step 3, increase space allocations, and redefine your catalog.

c. Remember to image copy the IMS catalog HALDB database after the load completes.

6. Start your IMS subsystem or subsystems.

**Enabling the IMS Catalog for Batch Processing**

1. Specify a parameter on the job EXEC statement that identifies the DFSDFxxx member used to extract information about the IMS catalog.

2. Use the optional DFS3CDX0 exit routine as an alternative to using the DFSDFxxx member to extract IMS catalog information. This might eliminate the need to modify existing batch JCL.

**Enabling IMS-Managed ACBs**

As IMS-managed ACBs require the IMS catalog, the steps for deploying the IMS catalog still apply, with the following differences:

1. To enable IMS-managed ACBs, when you populate the IMS catalog using the IMS catalog populate utility (DFS3PU00)
   a. Use the provided IMS catalog load PSB DFSCPL00.
   b. Add the control statement ‘MANAGEDACBS=SETUP’.

2. Modify the DFSDFxxx in PROCLIB – Add ACBMGMT=CATALOG as follows:
   ```
   <SECTION=CATALOG>
   CATALOG=Y,
   ALIAS=DFSC,
   ACBMGMT=CATALOG
   ```
   NOTE: In the `<catalog_section>`, ACBMGMT=ACBLIB is required to do the initial loading of the IMS directory. This is the default if you do not specify a value. Not using ACBMGMT=ACBLIB during the initial loading of the IMS directory will result in a U0071 abend.

3. If the Common Service Layer (CSL) is not currently deployed in your IMS environment, you must install and configure the CSL.

4. The SCI and Operation Manager (OM) address spaces are required. Update your PROCLIB with the SCI and OM JCL decks.

**CATALOG CONSIDERATIONS**

The sections that follow describe the things you must consider while enabling the IMS catalog.

**Shared IMS Catalog Versus Non-Shared IMS Catalog**

The following two processes detail the differences in using single-shared versus a non-shared IMS catalog.

- **IMSPLEX with multiple IMS subsystems using a single-shared IMS catalog:**
  - Only a single DFSDFxxx PROCLIB member is required with no ALIAS name defined.
  - A single occurrence of the IMS catalog exists in the DBRC RECON for the IMSPLEX.
  - ACBLIBs for IMS subsystems should contain identical content for any shared IMS database (DMB) or application control block (ACB). If all systems share all IMS application objects, you can share ACBLIBs. If all IMS application objects are not shared, you might want to consider using different ACBLIB data sets in one or more IMS subsystems.
  - If you are using a DFS3CDX0 exit, only a single exit for the IMSPLEX is required.

- **IMSPLEX with multiple IMS subsystems using non-shared IMS catalog:**
  - The DFSDFxxx PROCLIB member will specify the ALIAS name of the IMS catalog. You can provide multiple `<SECTION=CATALOGssid>` entries (each with an ALIAS identifier) or you can separate each IMS catalog alias into a unique DFSDFxxx member, associating each IMS subsystem to the appropriate DFSDFxxx member.
  - Each IMS catalog ALIAS HALDB definition resides in the DBRC RECON for the IMSPLEX.
  - ACBLIBs for all IMS subsystems contain identical content for any shared IMS database (DMB) or application control block (ACB). If all systems share all IMS application objects, you can share ACBLIBs. If all IMS application objects are not shared, you might want to consider using different ACBLIB data sets in one or more IMS subsystems.
  - If you are using a DFS3CDX0 exit, the exit parameter list includes the address of the IMS SCD and the IMS SSID. You can code a single exit that can accommodate all your non-shared IMS catalog parameters.
Sharing the IMS Directory Between IMS Systems on Different LPARS

If you share the IMS catalog and IMS directory across multiple IMS systems and running on different LPARs, appropriate action will be needed for the IMS directory because it is a PDSE data set:

- PDSE data sets and members can be shared if they are allocated with DISP=SHR.
- Multiple writers and readers can share the PDSE directory, and a single writer or multiple readers can each share the PDSE member.
- Any number of systems can have the same PDSE opened for input. If one system has PDSE opened for output, that PDSE can be opened on other systems only if the system uses the PDSE Extended sharing option. You can define the PDSE Extended sharing option by setting the PDSESHARING keyword in the 'IGDMSxx' SYS1.Parmlib member.
- The default is PDSESHARING(NORMAL), which supports data-set-level sharing only between multiple systems.
- PDSE data sets can be locked by either readers or by one system exclusively. Only readers allow the systems to perform any input operation but no output operation. The Exclusive option allows one system to perform input or output, but only that system can access the PDSE. IMS will experience an S213-70 abend when the IMS or DELTA PLUS product attempts to open data sets for update when the same IMS directory data sets are already opened for input on another IMS running on a different LPAR.
- If both IMS systems are running on the same LPAR, IMS or DELTA PLUS do not have this issue, and the S213-70 abend does not occur.
- PDSESHARING(EXTENDED) allows multiple systems to open the PDSE for both input and output. The open option could be blocked in cases where an UPDATE would cause data integrity issues, and this refers to updating the same member in the PDSE from multiple systems. Multiple systems in the same Sysplex that are running in extended mode can concurrently allow PDSE access for both input and output. PDSE sharing is supported within the same Sysplex, but not supported between different Sysplexes. PDSESHARING(EXTENDED) is highly recommended to avoid S213-70 abends.
- All systems that share the same IMS directory must operate in the same sharing mode (NORMAL or EXTENDED). To prevent damage to a shared PDSE, the operating system negotiates sharing rules when a system joins the Sysplex.

Deploying IMS-Managed ACBs

IMS-managed ACBs introduce a new component called the IMS directory to the IMS catalog implementation. The IBM statement of direction is that the IMS managed ACBs will be required in the future for the IMS online environment. IMS-managed ACBs were originally implemented in IMS 14. IMS managed ACBs replace the ACBLIBA/ACBLIBB that the IMS online environment used, and ACBLIB that the IMS batch environment used with the active directory. When IMS-managed ACBs are enabled for IMS online or batch environments, IMS loads the PSBs and DMBs from the directory.

The Catalog Populate utility will copy the PSBs and DBDs from ACBLIB to the directory for PSBs and DBDs that were inserted into the IMS catalog:

- IMS-managed ACBs require the implementing of the IMS catalog.
- The IMS CATALOG and the IMS directory component have a one-to-one relationship.
- The IMS directory is a replacement for the IMS ACBLIB libraries. The content of the IMS directory is built from content that is loaded into the IMS catalog.
- Currently, there are two directory data sets with IMS-managed ACBs. IMS concatenates them when allocating the directory. Many installations concatenate multiple ACBLIBs in IMSACB or ACBLIBA/ACBLIBB. The Catalog Populate utility can take multiple ACBLIBs as an input when populating the catalog and the directory. If there are duplicates and the ACBLIBs are concatenated to IMSACB01, the utility only uses the first one that is read. If you specify multiple ACBLIBs by using IMSACB01, IMSACB02, and so on, then ACBs from each IMSACBxx get inserted into the catalog unless the ACBs have the same timestamp, in which case only one gets inserted. The directory has the ACB with the latest timestamp.
- IMS will build and manage the IMS directory data sets (more than one data set, plus a staging IMS directory data set) as follows:
  - By default, the high-level qualifier (HLQ) of the IMS catalog database determines the prefix for the IMS directory data sets. The suffixes for the IMS directory data sets are:
    - DI plus a numeric suffix that IMS assigns for the IMS directory data sets
    - BSDS for the IMS directory bootstrap data set
    - STG for the IMS directory staging data set
• IMS determines the IMS catalog HLQ from the HALDB DSNPRREFX of the HALDB DBRC partition definition.

• You can use a different data set name prefix for the IMS directory and its associated data sets by specifying the parameter SYSDSHLQ=dsnprefx using the TYPE=CATDSHLQ MDA member when you run the IMS Catalog Populate utility (DFS3PU00). Each catalog alias has its own CATDSHLQ MDA member by specifying the alias name (xxxx) on the DDNAME=xxxxHLQ parameter to create the MDA member name.

• GSAM requires special processing because GSAM DBDs are not in ACBLIB, nor are GSAM DBPCBs in the PSB in ACBLIB. The Catalog Populate utility invokes a special ACBGEN that builds the GSAM DBDs and PSBs with GSAM in the directory. If the ACBCATWK DD statement is not present, the Catalog Populate utility builds all GSAM DBDs and PSBs (with DBPCBs) with GSAM.

• DOPT PSBs, when using ACBLIB, are in a dataset concatenated to ACBLIB in both ACBLIBA and ACBLIBB. They are not subject to the Online Change process. With IMS-managed ACBs, the DOPT PSBs are loaded from the active directory, so that the DOPT PSBs are written to the active directory without going through the staging directory. The IMS catalog does this in a similar way as when using a concatenated DOPT ACBLIB. Procedures for DOPT PSBs will need to be changed to use the Catalog Populate utility with MANAGEDACBS=(UPDATE,LATEST,SHARE) to copy the DOPT PSB to the active directory being used by the IMS online environment.

• IMS-managed ACBs eliminate ACBLIBA, ACBLIBB, and Online Change for ACBLIB. Procedures for copying the staging ACBLIB to the inactive ACBLIBA or ACBLIBB data set for Online Change or procedures for member Online Change for ACBLIB will have to be changed to use the Catalog Populate utility and either the IMPORT process or DELTA PLUS.

Catalog Out of Sync

Java SQL access to IMS databases uses a Get Unique Record (GUR) call to retrieve the catalog database record that is associated with the active ACBLIB member. The ACBGEN timestamp retrieves the database record.

If you do not run the Catalog Populate utility after an ACBGEN, the catalog will be out of sync with ACBLIB, and the GUR call will not retrieve the database record for the timestamp being used in ACBLIB. This will prevent Java SQL access to the database.

Directory Out of Sync

PSBLIB and DBDLIB can be out of sync with ACBLIB for batch. ACBLIB can be out of sync with the active ACB library for IMS online. With IMS-managed ACBs, the directory must be in sync with ACBLIB and the catalog.

The Catalog Populate utility uses the ACBGEN timestamp in the ACBLIB PDS directory while creating the PSB/DBD in the catalog. The active and staging timestamp is also in the HEADER segment in the catalog. The utility copies the ACBLIB member to the active directory or staging directory.

The timestamp in ACBLIB, the catalog, and the active or staging directories can get out of sync if you do not run the Catalog Populate utility after a successful ACBGEN. There are also potential out of sync possibilities when backing out a change by using the Catalog Purge utility.

Catalog Purge Utility

Unlike ACBLIB, the IMS catalog keeps a copy of each instance of the PSB and DBD that the Catalog Populate utility inserted into the catalog until the catalog is purged. Multiple instances are required for database versioning, or when the PSB/DBD has been inserted by the Catalog Populate utility but is not in the active ACBLIB yet.

The Catalog Purge utility deletes obsolete DBDs and PSBs instances from the catalog or deletes the entire DBD and PSB from the catalog.

To determine whether to delete an instance of the PSB or DBD, the Catalog Purge utility’s ANALYSIS mode uses the instances and Retention days in the HEADER segment for the PSB or DBD in the catalog. The actual deletion occurs only when the utility is in the PURGE mode.

The instances and retention days is set in the HEADER segment in the catalog based on the instances and retention days in the DFSDFxxx PROCLIB member. You can change the instances and retention days by using the Catalog Purge utility’s UPDATE statement.

An instance with the value of 1 would keep only the active PSB or DBD in the catalog—but be careful. The Catalog Populate utility creates an instance each time it is run with an updated ACBLIB member in the staging ACBLIB. If the value of retention days is less than the number of days the active instance has been in the catalog, the active PSB or DBD will be deleted.

Database versioning requires that non-active, previous instances of the DBD be kept in the catalog for as long as they are needed.
Migrating from IMS 14 to IMS 15

Migrating to a new IMS version has always required an ACBGEN BUILD PSB=ALL to generate an ACBLIB for the new IMS version that is not compatible with earlier versions. When using a catalog shared between the IMS versions, the ACBGEN results in the Catalog Populate utility inserting a new instance of each PSB and DBD in ACBLIB into the catalog, increasing the size of the catalog.

For IMS 14 installations that use IMS-managed ACBs, an ACBGEN is not mandatory when migrating to IMS 15. IMS 15 batch and online can read an IMS 14 directory, and IMS 15 batch can read an IMS 14 ACBLIB. IMS 14 can also read an IMS 15 directory and ACBLIB.

IBM continues to require that an ACBGEN be done if you are migrating to IMS 15 and are not using IMS-managed ACBs.

Catalog API

IMS provides a Catalog API DFS3CATQ that can be used to read the PSB/DMBs from the catalog in PSBLIB, DBDLIB, or ACBLIB format. This API reads the PSB/DBD from the active or staging directory, and not from the IMS catalog database. DFS3CATQ is documented in the IMS System Programming APIs manual.

IMS uses the Catalog API in the following scenarios:

• DBRC registration reads the DBD when CATALOG(name) is defined in the RECON or on the DBRC command.
• GSAM initialization needs to read the GSAM DBD from the directory.

Applications that need to read a PSB or DBD from the active or staging directory in PSBLIB, DBDLIB, or ACBLIB format can also use the Catalog API.

IMS is enhancing the Catalog API to allow the active as well as the staging directory to be opened concurrently. The original implementation allowed only one directory to be open at a time.

Only one Catalog API token can be open at the same time.

ADDRESSING THE IMS CATALOG CHALLENGES WITH A BMC SOLUTION

BMC is adding the Catalog Manager for IMS™ to address many of the challenges introduced with the IMS catalog and IMS managed ACBs.
Catalog Manager for IMS

The Catalog Manager for IMS is a no-charge offering that allows customers of BMC products for IMS to view IMS catalog contents, report on the control block information in the IMS catalog, and create jobs to perform DBD, PSB, and ACB GENs to populate the catalog.

Catalog Viewing

The Catalog Manager for IMS viewing capability helps you understand what information is loaded into your catalog by selecting segments in the catalog and displaying their contents. The information is shown in a tabular format with descriptive column headings. Hover over any column heading to see a detailed pop-up description of that column and its format. You can sort a column to view information in alternate sequences.
Catalog Reports

To help you verify that the information in the catalog is current and correct, Catalog Manager for IMS provides extensive integrity checking, including:

- **DBDAUDIT report** that displays information about your DBD definitions and flags inconsistencies such as:
  - A DBDGEN date later than the ACBGEN date
  - An ACB without a corresponding DBD
  - IMS release differences in the ACBLIB and DBDLIB
- **Cross-reference control block reports** that provide a variety of information about the relationships between the input control blocks, such as the names of PSBs that refer to a DBD
- **Compare reports** that flag the added, changed, or deleted source statements
- **DBD reports** that provide detailed information about the DBD control block

JCL Generation

Catalog Manager for IMS helps you load, update, and purge information in your IMS catalog. You can use an interactive interface to generate PSB, DBD, and ACB jobs to update these libraries and populate the IMS catalog. You can also use this function to perform a standalone catalog populate without the need to perform an ACBGEN and create jobs to purge unneeded catalog entries.
**IMS Command Facility**

If you need to issue IMS commands during catalog management, you can issue them and view the results from within Catalog Manager for IMS. You do not need to switch to a different product simply for IMS command processing.

**Integration with the DBA Toolkit**

Catalog Manager for IMS is easy to activate and use due to its integration into the existing DBA Toolkit console and workflow. Once installed, the Catalog Manager for IMS functions become additional tabs within the DBA Toolkit and use the same infrastructure.

**Future Direction for Catalog Manager for IMS**

BMC has long been a partner with our IMS customers, providing products and assistance with each new release of IMS. We will continue this focus with the IMS catalog and directory developments. Our goal is to provide incremental improvement and enhancements to the Catalog Manager for IMS based on the feedback from our users, and to build the best solution for catalog and directory management in the industry.

---

**FOR MORE INFORMATION**

To learn more about Catalog Manager for IMS, visit [bmc.com/it-solutions/database-management-ims](http://bmc.com/it-solutions/database-management-ims)

---

**ABOUT THE AUTHORS**

**Robert Blackhall**

Robert Blackhall is an IMS architect with over 20 years of software development experience at BMC. His work includes architecture of the Message Advisor for IMS product and software development for Delta/IMS and other IMS transaction-based products. Robert has a bachelor’s degree in computer science from Wichita State University and master’s degree in management from Webster University.

**Duane Wente**

Duane Wente is an IMS consultant with extensive BMC experience. He has helped many large BMC customers over the last 20 years understand, implement, and successfully use BMC IMS solutions. He has worked with accounts on IMS discovery, disaster recovery, database performance, and tuning engagements. Duane has a bachelor’s in management information systems from the University of Houston.

BMC is a global leader in innovative software solutions that enable businesses to transform into digital enterprises for the ultimate competitive advantage. Our Digital Enterprise Management solutions are designed to fast track digital business from mainframe to mobile to cloud and beyond.