Best practices
Creating, storing, and managing definitions for IMS resources

Sandy Stoob
IMS Systems Development

David Hanson
IMS Information Development

Anu Vakkalagadda
IMS Systems Development

Janna Mansker
IMS Systems Development

Issued: September, 2013
Creating, storing, and managing definitions for IMS resources

Executive Summary

Introduction

History of IMS resource definition and management systems

Use resource naming conventions

Store and back up resource definitions

Where to store resource definitions

When to back up resource definitions

How to back up resource definitions

Manage resources with Operations Manager (OM), Single Point of Control (SPOC), and type-2 commands

Automate resource definition and management

Define resources and data sets dynamically

Drawbacks of defining resources dynamically

Benefits of defining resources dynamically

Define and manage resources consistently across an IMSplex

Plan for resource errors before they happen

Best practices

Conclusion

Further reading

Contributors

Notices

Trademarks

Contacting IBM
Executive Summary

This article introduces the IBM Information Management System (IMS) resource definition history, describes general best practices for defining, storing, and managing resources, and provides a guide to other articles that each describe best practices for a specific type of IMS resource.
Introduction

When IMS was first introduced in 1968, the only way to define resources was the IMS system definition process. Traditional system definition is a multi-step process that involves running utilities to generate resources (also known as gens) including the stage 1 system generation, database definition generation (DBDGEN), and program specification block generation (PSBGEN). The IMS system definition stage 1 process involves coding IMS Assembler macros to define IMS resources and running the z/OS High Level Assembler program to compile them into output that becomes input into the stage 2 process. The DBDGEN and PSBGEN process both include two steps: first, coding IMS Assembler macros to define IMS database or program; and second, running the DBDGEN or PSBGEN utility to assemble the macro definition into a database or program definition that can be used by IMS. In subsequent years, more utilities were added to generate definitions, such as the Application Control Block generation utility (ACBGEN). In the traditional system definition process, IMS has to be shut down before new resource definitions are incorporated. IMS is unavailable to do work while it is shutdown. As more and more enterprises need to operate 24x7 with fewer planned outages, and users need to be able to easily define large numbers of resources, the traditional system definition method is no longer adequate.

There are now several methods for defining, storing, and managing IMS resources. Over time, IMS has moved toward a model of resource definition and management that is more dynamic, more capable of expansion to large environments with many IMS systems, and more readily managed with centralized repositories. However, there are trade-offs between the older, static methods of resource definition and the newer dynamic methods that are now available. If the IMS environment is small (for example, with a single IMS region) and there are no plans for future growth, using the original IMS system definition process is probably the best option. However, this would require you to do a sysgen as you migrate to a new IMS release. One or more of the newer methods will provide easy migration to new releases. If there are numerous IMS systems in an IMSplex with more systems and resources being added frequently, one or more of the newer methods will provide benefits that outweigh the overhead of implementing the new methods for resource definition. The best practices described in this article are guidelines to help you determine the best path for your unique environment.

History of IMS resource definition and management systems

The following table shows a brief overview of the different enhancements to resource definition and management in IMS over its history. This table is not meant to provide an exhaustive list, or a complete description for all of the features listed.
<table>
<thead>
<tr>
<th>Year</th>
<th>Release</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1984</td>
<td>1.3</td>
<td>Online Change (OLC)</td>
<td>OLC is an IMS function that supports adding, changing, and deleting database, DMB, MFS format, program, PSB, routing code, and transaction resources online. OLC can make changes to the ACBLIB, MODBLKS, and the FMTLIB.</td>
</tr>
<tr>
<td>1989</td>
<td>3.1 or earlier</td>
<td>Dynamic Option (DOPT) PSBs</td>
<td>A DOPT program is loaded from the ACBLIB each time the program is scheduled, without requiring an online change. When a change is made to DOPT PSB, it is made active in the system immediately the next time that the program is scheduled. For more information, see Best practices for defining and managing DOPT and GPSB PSB resources</td>
</tr>
<tr>
<td>1993</td>
<td>4.1</td>
<td>Generated PSBs (GPSB)</td>
<td>GPSBs are generated by IMS without the need for a PSBGEN operation. For more information, see Best practices for defining and managing DOPT and GPSB PSB resources</td>
</tr>
<tr>
<td>1993</td>
<td>4.1</td>
<td>Extended Terminal Option (ETO)</td>
<td>ETO allows you to add VTAM terminals and users to your IMS system without predefining them during system definition.</td>
</tr>
<tr>
<td>1993</td>
<td>4.1</td>
<td>DFSINSX0 exit</td>
<td>The Destination Creation Exit (DFSINSX0) is introduced and can be used to dynamically create a logical terminal.</td>
</tr>
<tr>
<td>1997</td>
<td>6.1</td>
<td>Online Change for Data Entry Databases (DEDBs)</td>
<td>OLC is enhanced to support adding, changing, and deleting Fast Path DEDBs from ACBLIB.</td>
</tr>
<tr>
<td>1997</td>
<td>6.1</td>
<td>DFSINSX0 exit creates transactions for queueing</td>
<td>The Destination Creation Exit (DFSINSX0) can be used to dynamically create a transaction (in a shared queues environment) for queueing only when a destination for a message does not exist.</td>
</tr>
<tr>
<td>2002</td>
<td>8.1</td>
<td>Common Service Layer (CSL)</td>
<td>The CSL is a set of address spaces that help manage IMS communications, operations, and resources in an IMSplex. New address spaces were added to manage the multiple IMS systems and resources in an IMSplex: • Structured Call Interface (SCI) • Operations Manager (OM) • Resource Manager (RM)</td>
</tr>
<tr>
<td>2002</td>
<td>8.1</td>
<td>Global Online Change (GOLC)</td>
<td>GOLC is an enhancement to OLC that coordinates and simplifies online change to add, change, and delete DMB, MFS format, program, PSB, routing code, and transaction resources between multiple IMS systems in an IMSplex at the same time.</td>
</tr>
</tbody>
</table>
| 2002 | 8.1     | OLCSTAT dynamic allocation | The OLCSTAT data set can be allocated dynamically, to minimize the risk that a long-running, multi-step batch job might prevent Global Online Change from making updates to
<table>
<thead>
<tr>
<th>Year</th>
<th>Release</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002</td>
<td>8.1</td>
<td>Sysplex Terminal Management (STM)</td>
<td>STM provides global terminal and command status for VTAM terminals.</td>
</tr>
<tr>
<td>2007</td>
<td>10.1</td>
<td>Dynamic Resource Definition (DRD)</td>
<td>DRD can dynamically create, update, query, and delete certain resource definitions (databases, application programs, Fast Path routing codes, and transactions) without using traditional system definition or OLC. Resource definitions are stored in an RDDS data set for coldstart.</td>
</tr>
<tr>
<td>2007</td>
<td>10.1</td>
<td>Serialized Program Management</td>
<td>Serialized program management allows users in a shared-queues environment to prevent application programs that are defined as serial from being scheduled in parallel on another IMS system in an IMSplex. Information about scheduled serial PSBs is maintained in the RM resource structure to ensure that the serial PSB is scheduled in only one IMS across an IMSplex at any point in time.</td>
</tr>
<tr>
<td>2007</td>
<td>10.1</td>
<td>ACB library member online change (ACBMBR OLC)</td>
<td>In an IMSplex environment, you can use the ACB member online change (OLC) function to add or change individual members of the ACB library directly into the active ACBLIB. Only the DMBs and PSBs affected by the online change are quiesced.</td>
</tr>
<tr>
<td>2007</td>
<td>10.1</td>
<td>DFSINSX0 exit</td>
<td>The Destination Creation Exit (DFSINSX0) can be used to dynamically create a transaction for scheduling when the destination for a message does not exist. Additionally, a program resource for the transaction can also be created. The transaction and program can be created at a single IMS or for multiple IMSs.</td>
</tr>
<tr>
<td>2007</td>
<td>10.1</td>
<td>Systems Management Global Status</td>
<td>IMS users can maintain global command status for database, area, or transaction resources in IMS Resource Manager that can be applied during IMS restart. Commands can now be synchronized across the IMSplex even if an individual IMS system was offline or unavailable when the command was issued.</td>
</tr>
<tr>
<td>2007</td>
<td>10.1</td>
<td>Type-2 commands for resource definition</td>
<td>The IMS type-2 command interface includes methods to define certain IMS resources. The IMS type-2 commands have been enhanced with each subsequent release to support more resource definition operations and more resource types.</td>
</tr>
<tr>
<td>2009</td>
<td>11.1</td>
<td>ACBLIB dynamic allocation</td>
<td>The ACB library data sets can be dynamically allocated. Using DFSMDA to dynamically allocate IMSACBA and IMSACBB allows you to re-size the inactive ACBLIB data sets, add data sets to the concatenation, or change data sets in the concatenation without shutting down your IMS system.</td>
</tr>
</tbody>
</table>
| 2011 | 12.1    | IMSRSC Repository | The IMS resource definition (IMSRSC) repository is a set of
VSAM key sequenced data sets (KSDSs) that are used for storing resource and descriptor definitions. The initial implementation is to store the IMS databases, transactions, programs, and routing codes and their descriptors. This repository is an alternative to the resource definition data set (RDDS) that is used with the dynamic resource definition (DRD) function.

The IMSRSC repository is a centralized store to store consistent resource definitions for all the IMS systems in the IMSplex and make them available for coldstart.

<table>
<thead>
<tr>
<th>Year</th>
<th>Version</th>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012</td>
<td>12.1</td>
<td>IMS catalog</td>
<td>The IMS catalog database contains trusted metadata about an IMS system, based on current and past versions of the ACBLIB and DBDLIB for that system. The catalog can also contain application metadata such as decimal data specifications (scale and precision), data structure definitions, and mapping information. The IMS catalog database can be queried with standard DL/I processing, with DL/I processing through the Universal DL/I driver, and with SQL through the Universal JDBC driver.</td>
</tr>
</tbody>
</table>

Table 1: History of IMS resource definition systems
Use resource naming conventions

Before defining resources, a best practice is to standardize the naming conventions that you will use for resource, data set, and subsystems to help manage the system, minimize errors, analyze problems, and make advantageous use of IMS command generic parameter support (also known as wildcard support). You can reserve one or more characters in the resource name to identify the resource type, resource location, application group, IMS version, IMSplex, or any other important resource attribute, so that you can get more information about the resource just by looking at the resource name.

Some examples of resource naming conventions:

- All IMS application programs start with “P”, to identify application program resources
- All IMS application programs start with a 2 or 3 character prefix that identifies the application group
- A data set name includes what type of data set it is, such as ACBLIB, FMTLIB, MODBLKS, or OLCSTAT
- A data set name includes the IMS version, to easily identify whether the data set name is current or back level.
- All IMS systems are named IMxx, where xx is a number or character, to easily determine that it is an IMS subsystem
- Data set names contain the letter “L” for local, “G” for global, or “S” for shared.

For example: the IMS version 12 ACBLIB data sets can be named IMSTESTS.V12.ACBLIBA and IMSTESTS.V12.ACBLIBB, where “IMSTEST” indicates an IMS test system, “S” indicates a shared ACBLIB, “V12” indicates IMS version 12, ACBLIBA indicates the ACBLIB “A” side, and ACBLIBB indicates the ACBLIB “B” side.

When you standardize on known naming conventions, you can use the wildcard support in IMS type-2 commands to your advantage to quickly identify the resources you want and make changes. Some examples:

- QUERY PGM NAME(P1*), where all application programs within one application begin with the characters “P1”, to display all application programs in the P1 application with one command.
- QUERY TRAN NAME(T%%1A*), where all transactions begin with the letter “T”, and the 4th and 5th characters identify the IMS suffix of the IMS where the transaction is routed.
- QUERY DB NAME(*L), where the last character identifies whether the database is local (L), or global (G), to display only local databases.
• QUERY TRAN NAME(%%%%%), where all 4-character transactions defined to IMS are CICS transactions to be sent to CICS using ISC.

See Type-2 command environment and IMS type-2 command format for more information.

**Store and back up resource definitions**
No matter what method is used to define resources, resource definitions should be backed up in case an error causes the resource definitions to be lost. To ensure that your known-good resource definitions are never permanently lost, determine where, when, and how to back them up.

**Where to store resource definitions**
This list contains some examples of storage locations for resource definitions. The storage locations used are based on the size and complexity of the system, the functions used, whether IMSplex is used, the number of resources defined, and other unique needs of the installation.

- IMSRSC repository
- RDDS (system or non-system)
- REXX SPOC program that executes CREATE commands
- Batch SPOC utility (CSLUSPOC) that executes CREATE commands
- IMS log records (Use the DFSURCL0 utility to create an RDDS based on log records)
- ETO descriptors
- DFSINSX0 exit routine
- The online IMS system
- ACBLIB dataset
- MODBLKS dataset
- FMTLIB dataset
- IMS system definition stage 1 input file
- Roll your own storage and back up solutions
When to back up resource definitions

Resource definitions can be backed up as frequently as needed by the installation. If losing resource definitions would be catastrophic for the installation, consider backing up resource definitions as frequently as possible. If losing resource definitions is not a problem, back up resource definitions less frequently or not at all. For example, in a cloned test environment, running back up operations can consume unnecessary system resources.

How to back up resource definitions

There are several z/OS and IMS utilities available to store resource definitions in another place or another format, as a backup or for other purposes. For example:

- **CSLURP10** utility: writes resource definitions to the IMSRSC repository from an RDDS
- **CSLURP20** utility: creates an RDDS from the IMSRSC repository
- **DFSUOCU0** utility (copies ACBLIB, MODBLKS, or FMTLIB members to another data set)
- **DFSURCP0** utility: creates an RDDS copy
- **DFSURCL0** utility: creates an RDDS from IMS log records
- **DFSURCM0** utility: creates an RDDS from a MODBLKS data set
- **DFSURDD0** utility (creates IMS system definition macro statements or CREATE commands)
- **IDCAMS**: copies Repository VSAM data sets to another data set
- **IEBCOPY**: copies members between one or more partitioned data sets, or partitioned data sets extended (PDSEs)
- **DFS3CCE0/DFS3CCI0** utilities: copies the IMS catalog database and resource library data sets from one IMS system to another
Manage resources with Operations Manager (OM), Single Point of Control (SPOC), and type-2 commands

It is a best practice to manage resources and IMS systems in an IMSplex using Operations Manager (OM), Single Point of Control (SPOC), and IMS type-2 commands. The benefits of OM and its associated functions include:

- More information and feedback about the state of resources in the IMSplex
- More informational completion code text and error text
- The OM audit log
- SPOC automation options
- SPOC scrolling, sorting, and paging
- CSLOMSUB to subscribe to unsolicited messages

The benefits of Operations Manager and type-2 commands often justify the associated cost of the OM address space even in single-system IMS configurations.

See [Overview of the Operations Manager](#) for more information.

Automate resource definition and management

Automating resource definition and management improves availability, because it takes less time to create, delete, manage, and update resources, which reduces the amount of time that those resources are unavailable. This is especially helpful for defining large numbers of resources. Additionally, automating your resource definition process allows you to re-use definition practices and procedures that are known to work, and it greatly reduces the chance of human error.

Two of the most accessible ways to automate your resource definition process are the batch SPOC and the REXX SPOC. Both the batch and REXX SPOC require OM.

See [REXX SPOC API](#) or [ Issuing Batch SPOC commands](#) for more information.

Define resources and data sets dynamically

There are now several methods for defining resources and data sets dynamically, rather than use the IMS system definition process. These include DOPT PSBs, Extended Terminal Option (ETO), GPSBs, Dynamic Resource Definition (DRD), OLCSTAT dynamic allocation, ACBLIB dynamic allocation, ACB library member online change,
and the IMSRSC repository. IMS continues to roll out new ways of defining resources dynamically in new IMS releases. It is a best practice to evaluate the costs and benefits of current and future methods of defining resources dynamically, in order to reap those benefits if they outweigh the costs.

**Drawbacks of defining resources dynamically**

- Cost of running additional IMS address spaces in your environment.
- Administrative overhead required to manage the new IMS address spaces.

**Benefits of defining resources dynamically**

- IMS remains available while resource definition changes are made.
- Decreases the number of utilities that need to be run.
- Only affected resources are quiesced.
- Easier to manage large numbers of resources and subsystems in an IMSplex.
- Create and delete resources as needed, which frees up CSA storage.
- Improved automation functions.
- Correct resource definition problems by deleting them and re-defining them.
- Manage resource definitions more easily with EXPORT, IMPORT, and QUERY type-2 commands.
- ETO resources are cleaned up automatically at /CLSDST time if ETO is enabled.
- Logical terminals (LTERMs), nodes, and users created dynamically by the DFSINSX0 exit routine are also cleaned up automatically at IMS checkpoints.

**Define and manage resources consistently across an IMSplex**

If you have two or more IMS systems in your IMSplex, or might have more than two IMS systems in the future, the resources need to be defined consistently in all of the IMS systems. If resources are not defined consistently across an IMSplex, problems can occur. For example, if a transaction is defined to one system as conversational, but to another...
system as non-conversational, transaction processing will not work correctly in the IMSplex. There are several methods for ensuring consistent resource definitions and status across an IMSplex:

- Manually
- Local online change separately on each IMS
- Global online change
- Systems management global status
- ACB library member online change
- Serialized program management
- IMSRSC repository

For a more complex IMSplex, consider defining an IMSRSC repository. You can define all DRD resources in one place and keep the resource definitions consistent across the IMSplex. DRD resources include databases, programs, routing codes, and transactions. Consider the trade-offs of managing resource definitions locally for each IMS in an IMSplex, versus managing resource definitions using the IMSRSC repository, with the added complexity of additional IMS address spaces and the IMSRSC repository in your IMSplex.

See [IMS repository function enhancements](#) for more information.

**Plan for resource errors before they happen**

Establish manual or automated procedures for handling error situations. The worst time to develop recovery procedures is during an outage.

By automating your resource error recovery operations, you can determine once what needs to be done in an error situation, then avoid the problem or quickly recover from it in the future. If an error occurs, you can automate issuing DISPLAY and QUERY commands and use the resulting status to determine what the automation should do about the error. The output of type-2 commands that are issued through the batch or REXX SPOC, or through the OM API, are intended to be parsed programmatically by automated operator programs. However, type-2 commands require OM.
Best practices

- Formulate and use a standardized set of naming conventions for your IMS resources
- Store and back up resource definitions
- Automate resource definition management
- Define resources dynamically when possible
- Define and manage resources consistently across an IMSplex
- Plan for resource errors before they occur
Conclusion

Throughout its history, IMS has continually moved away from the model of statically defined resources for individual IMS systems and toward a model of resource management that is dynamic, centralized, and automated. This article gives an overview of these methods and provides links to other articles on methods and best practices for specific IMS functions, to help you make informed decisions about the best ways to define, store, and manage these resource definitions in your environment.
Further reading

- Best practices: Creating, storing, and managing definitions for DOPT PSB and GPSB resources


Contributors

Steve Nathan
IMS Level 2 Support

Skip Gingras
IMS Level 2 Support

Tom Morrison
IMS TM Development

Kevin Stewart
IMS Level 2 Support
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. 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. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

Without limiting the above disclaimers, IBM provides no representations or warranties regarding the accuracy, reliability or serviceability of any information or recommendations provided in this publication, or with respect to any results that may be obtained by the use of the information or observance of any recommendations provided herein. The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any recommendations or techniques herein is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Anyone attempting to adapt these techniques to their own environment do so at their own risk.

This document and the information contained herein may be used solely in connection with the IBM products discussed in this document.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

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

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE: © Copyright IBM Corporation 2013. All Rights Reserved.

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

**Trademarks**

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml

Windows is a trademark of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

**Contacting IBM**

To provide feedback about this paper, write to imspubs@us.ibm.com

To contact IBM in your country or region, check the IBM Directory of Worldwide Contacts at http://www.ibm.com/planetwide

To learn more about IBM Information Management products, go to http://www.ibm.com/software/data/