IMS

Operations Guide

Version 9

IMS

Operations Guide

Version 9

ZES1-2355-00

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 383.

Quality Partnership Program (QPP) Edition (December 2003) (Softcopy Only)

This QPP edition applies to Version 9 of IMS (product number 5655-J38) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

About This Book

This softcopy book is available only in PDF and BookManager formats. This book is available on the IMS Version 9 Licensed Product Kit (LK3T-7213). To get the most current versions of the PDF and BookManager formats, go to the IMS Web site at www.ibm.com/ims and link to the Library page.

This book describes the task of operating an IMS system and the administrative task of planning for IMS operations and recovery. This book is written for IMS operators, system programmers, and database administrators who are responsible for design, operation, and recovery procedures for their installation.

Although batch considerations are discussed, this book is oriented primarily towards online operations. The term “online” applies to all IMS environments: DB/DC, DBCTL, and DCCTL.

Summary of Contents

This book is divided into several parts, each written with a different reader in mind:

- Part 1, “IMS Operations: Concepts and Tools,” on page 1 introduces the concepts and tools described throughout this book. This part is intended for all readers.
- Part 2, “Operating an IMS Subsystem,” on page 21 describes the task of operating IMS. The focus of this part is on understanding IMS operations and designing operating procedures. This part is intended for IMS system programmers.
- Part 3, “Developing Operating Procedures,” on page 305 describes the types of procedures that you need to develop to operate IMS, who uses them, and the type of information that you must develop for your end users.
- Appendix A, “IMS Support of Devices,” on page 355 describes IMS support for various devices.
- The “Bibliography” on page 387 contains a list of the publications listed in this book.

When this book uses the word “you,” it is referring to the primary reader of each part.

Prerequisite Knowledge

IBM® offers a wide variety of classroom and self-study courses to help you learn IMS. For a complete list, see the IMS home page on the World Wide Web at: www.ibm.com/ims.

...For Operators

Before using this book, you should understand:

- Basic IMS concepts
- Your installation’s IMS system

...For System Programmers

Before using this book, you should understand:

- Basic IMS concepts
- The IMS environment

- Your installation’s IMS system
- Administration of the IMS system and databases
- MVS™ control programs
- VSAM Access Method Services

How to Read Syntax Diagrams

Each syntax diagram in this book begins with a double right arrow and ends with a right and left arrow pair. Lines that begin with a single right arrow are continuation lines. You read a syntax diagram from left to right and from top to bottom, following the direction of the arrows.

Conventions used in syntax diagrams are described in Table 1:

Table 1. How to Read Syntax Diagrams

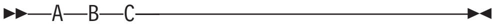
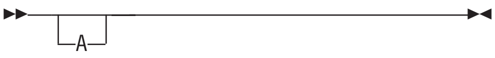
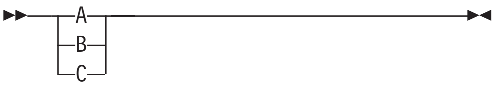
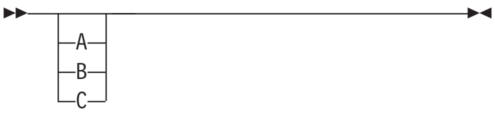
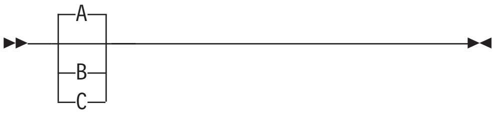
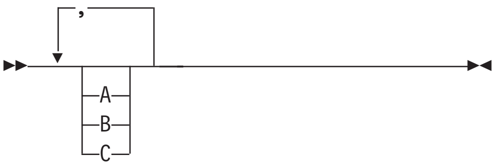
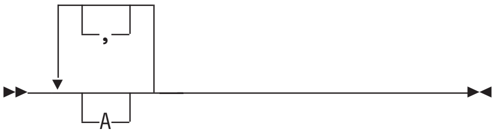
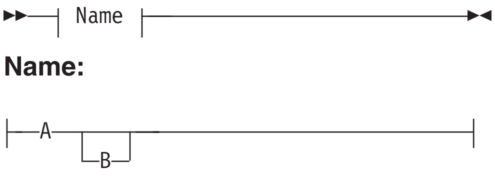
Convention	Meaning
	You must specify values A, B, and C. Required values are shown on the main path of a syntax diagram.
	You have the option to specify value A. Optional values are shown below the main path of a syntax diagram.
	You must specify value A, B, or C.
	You have the option to specify A, B, C, or none of these values.
	You have the option to specify A, B, C, or none of these values. If you don’t specify a value, A is the default.

Table 1. How to Read Syntax Diagrams (continued)

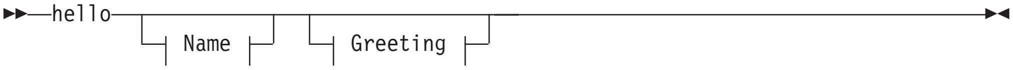
Convention	Meaning
	You have the option to specify one, more than one, or none of the values A, B, or C. Any required separator for multiple or repeated values (in this example, the comma) is shown on the arrow.
	You have the option to specify value A multiple times. The separator in this example is optional.
	Sometimes a diagram must be split into fragments. The syntax fragment is shown separately from the main syntax diagram, but the contents of the fragment should be read as if they are on the main path of the diagram.
Punctuation marks and numbers	Enter punctuation marks (slashes, commas, periods, parentheses, quotation marks, equal signs) and numbers exactly as shown.
Uppercase values	Keywords, their allowable synonyms, and reserved parameters, appear in uppercase letters for z/OS. Enter these values exactly as shown.
Lowercase values without italics	Keywords, their allowable synonyms, and reserved parameters, appear in lowercase letters for UNIX. Enter these values exactly as shown.
Lowercase values in italics (for example, *name*)	Supply your own text or value in place of the *name* variable.
ƀ	A ƀ symbol indicates one blank position.

Other conventions include the following:

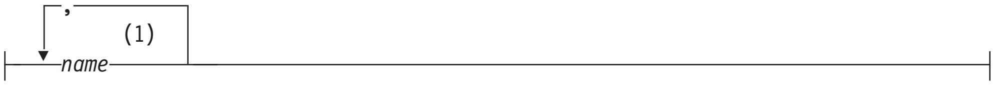
- When entering commands, separate parameters and keywords by at least one blank if there is no intervening punctuation.
- Footnotes are shown by a number in parentheses, for example, (1).
- Parameters with number values end with the symbol #.
- Parameters that are names end with ’name’.
- Parameters that can be generic end with the symbol *.

Syntax Diagram Example

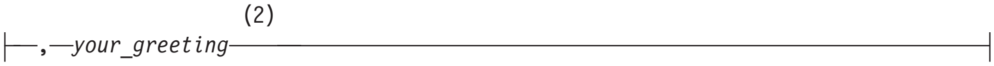
Here is an example syntax diagram that describes the **hello** command.



Name:



Greeting:



Notes:

1 You can code up to three names.

2 Compose and add your own greeting (for example, how are you?).

According to the syntax diagram, these are all valid versions of the **hello** command:

```
hello
hello name
hello name, name
hello name, name, name
hello, your_greeting
hello name, your_greeting
hello name, name, your_greeting
hello name, name, name, your_greeting
```

The space before the *name* value is significant. If you do not code *name*, you must still code the comma before *your_greeting*.

How to Send Your Comments

Your feedback is important in helping us provide the most accurate and highest quality information. If you have any comments about this book or any other IMS information, you can do one of the following:

- Go to the IMS home page at: www.ibm.com/ims. There you will find an online feedback page where you can enter and submit comments.
- Send your comments by e-mail to imspubs@us.ibm.com. Be sure to include the name of the book, the part number of the book, the version of IMS, and, if applicable, the specific location of the text you are commenting on (for example, a page number or table number).

Change Indicators

Technical changes are indicated in this publication by a vertical bar (|) to the left of the changed text.

Summary of Changes

Changes to This Book for IMS Version 9

This edition is a draft version of this book intended for use during the Quality Partnership Program (QPP). Contents of this book are preliminary and under development.

New information on the following enhancements is included:

- New commands throughout the book and in the new Appendix B, "IMS Type-1 and Type-2 Command Equivalents," on page 381. These new commands are:
  - QUERY AREA
  - QUERY DB
  - UPDATE AREA
  - UPDATE DATAGRP
  - UPDATE DB
- VTAM Multinode Persistent Session (MNPS) as a replacement of XRF USERVAR, in "Warm Start" on page 89, "Emergency Restart" on page 90, and in Chapter 15, "Extended Recovery Facility (XRF)," on page 237
- Command authorization for DBRC commands submitted as online (/RM) commands, in "Issuing DBRC Commands" on page 103.
- Support for type-2 commands in IMSplexes without the use of RM, in Chapter 10, "Making Online Changes," on page 121.
- RACF enhancements to replace the Security Maintenance utility (SMU), in the following sections:
  - "Changing the System Definition Online" on page 123
  - "Installing TCO" on page 186
  - "Using Type-1 AOI" on page 198
  - "Using Type-2 AOI" on page 200
- A new section, "Global Online Change after XRF Takeover and DBCTL Standby Emergency Restart" on page 127
- Changed recovery recommendations for shared VSO areas, in Chapter 12, "Recovering from IMS Failures," on page 153
- Support for Knowledge-Based Log Analysis (KBLA), in "Recovering a Log" on page 158 and "Using IMS System Log Utilities" on page 95
- Support for HALDB Online Reorganization with restrictions documented in Chapter 3, "Commands for IMS Tasks," on page 29, Chapter 15, "Extended Recovery Facility (XRF)," on page 237, and Chapter 16, "Remote-Site Recovery (RSR)," on page 265
- The /START DATABASE command, which now activates online forward recovery (OFR), eliminating the need to run multiple /START AREA commands. See Chapter 16, "Remote-Site Recovery (RSR)," on page 265.

Library Changes for IMS Version 9

Changes to the IMS Library for IMS Version 9 include the addition of new titles, the change of one title, and a major terminology change.

New and Revised Titles

The following list details the major changes to the IMS Version 9 library:

- *IMS Version 9: HALDB Online Reorganization Guide and Reference*

  The library includes a new book: *IMS Version 9: HALDB Online Reorganization Guide and Reference*. This information is available only in PDF and BookManager formats.
- *IMS Version 9: An Introduction to IMS*

  The library includes a new book: *IMS Version 9: An Introduction to IMS*.
- The book formerly titled *IMS Version 8: IMS Java User's Guide* is now titled *IMS Version 9: IMS Java Guide and Reference*.

Terminology Changes

IMS Version 9 introduces new terminology for IMS commands:

type-1 command
A command, generally preceded by a leading slash character, that can be entered from any valid IMS command source. In IMS Version 8, these commands were called *classic* commands.

type-2 command
A command that is entered only through the OM API. Type-2 commands are more flexible and can have a broader scope than type-1 commands. In IMS Version 8, these commands were called *IMSplex* commands or *enhanced* commands.

Accessibility Enhancements

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products. The major accessibility features in z/OS products, including IMS, enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

User Assistive Technologies

Assistive technology products, such as screen readers, function with the IMS user interfaces. Consult the documentation of the assistive technology products for specific information when you use assistive technology to access these interfaces.

Accessible Documentation

Online information for IMS Version 9 is available in BookManager format, which is an accessible format. All BookManager functions can be accessed by using a keyboard or keyboard shortcut keys. BookManager also allows you to use screen readers and other assistive technologies. The BookManager READ/MVS product is included with the z/OS base product, and the BookManager Softcopy Reader (for workstations) is available on the IMS Licensed Product Kit (CD), which you can download from the Web at www.ibm.com.

Keyboard Navigation of the User Interface

Users can access IMS user interfaces using TSO/E or ISPF. Refer to the *z/OS V1R1.0 TSO/E Primer*, the *z/OS V1R1.0 TSO/E User's Guide*, and the *z/OS V1R1.0 ISPF User's Guide, Volume 1*. These guides describe how to navigate each interface, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Part 1. IMS Operations: Concepts and Tools

Chapter 1. Introduction to IMS Operations and Recovery

For any large system, operations and recovery go together: *operations* refers to the day-to-day activities involved in keeping the system running smoothly, and *recovery* refers to the activities required to bring a failed system back online. A book about operations would be incomplete if it did not describe what you need to do for recovery, when the system (or some part of it) runs into problems.

Another important task related to operations is planning for both day-to-day operations and for recovery. Part of the task of planning is developing procedures that specify the roles of the IMS™ operators and system programmers. These procedures clarify the responsibilities of each of these roles.

This chapter introduces both operations and recovery for IMS and describes some of the tools provided by IMS and MVS for both tasks.

Understanding the Operations Task

The task of planning for operations has two major parts:

- Selecting functions and tools
- Developing operating and end-user procedures

First, you must decide how to use the tools IMS provides for operating your system. This includes choosing, for example, whether to use dual logging when setting up your log, how often to make backup copies of your database, and whether to use DBRC to control recovery of databases.

Second, you must develop procedures for operating and using IMS. Operating procedures must tell operators how to:

- Start and restart IMS
- Control IMS
- Make online changes to modify IMS
- Shut down IMS
- Run various IMS utilities
- Recover from IMS and other failures

These procedures are primarily for the master terminal operator (MTO), who operates IMS from the master terminal. They are secondarily for the people who assist in keeping IMS running smoothly, such as the recovery specialist who works with many of the more complex IMS recovery problems.

End-user procedures tell end users how to:

- Operate their terminal
- Establish a connection to IMS
- Communicate with IMS (using their specific applications)
- Terminate a connection to IMS
- Respond to any error conditions they encounter

This book is not an exhaustive treatment of operations. More detailed information on various aspects of operating your system is provided in other IMS books.

Related Reading:

- *IMS Version 9: Command Reference* gives guidance level information and detail about the commands that are used to operate IMS.
- *IMS Version 9: Utilities Reference: Database and Transaction Manager* and *IMS Version 9: Utilities Reference: System* give guidance level information and detail about the various utilities that are available with IMS. *IMS Version 9: DBRC Guide and Reference* includes information about the Database Recovery Control utility, which processes DBRC commands.
- *IMS Version 9: Administration Guide: System* gives guidance level information about how to set up and administer an IMS system.

Understanding the Recovery Task

This section introduces basic IMS recovery concepts and tools to provide a framework for the more detailed information in later chapters. Recovery is the largest and most complex part of operations. For this reason, any discussion of operations is primarily a discussion of recovery.

What Is Recovery?

Recovery is the task of restoring a failed system to normal operations. Recovering a system can involve:

- Databases
- User requests to process data
- Programs that do the processing (application programs)
- Output sent to users

A recoverable system ensures:

- Data is not lost
- Incomplete changes to a database are not saved

Data can be lost in two ways. It can be physically lost—the disk or tape on which it resides can be damaged or misplaced. Or it can be “logically” lost—the data becomes incorrect or loses its relationship to other data. A system that ensures *data integrity* ensures that data cannot be lost or saved incompletely.

Example: A System without Recovery

To show how things can go wrong, consider a hypothetical system that has no recovery mechanisms. A user requests a transaction that handles a customer order. Part of the application program’s work is to:

- Decrease the stock-on-hand number in the database
- Add the cost to the customer’s outstanding bill
- Add the amount to a daily-total-sales accumulator

Effects of System Failure

What if the system fails between the time the stock-on-hand number is decreased and the charges are allocated? If the user thinks the transaction is complete, the customer bill record and the total sales record in the database will be incorrect. If the user thinks the transaction failed and reruns it, the stock-on-hand number will be decreased a second time, and that record will also be wrong. In either case, data integrity has been lost.

Effects of Program Abend

What if the program begins changing records in the database and then terminates abnormally (abends)? The result is the same as the result described in “Effects of System Failure” on page 4: a half-changed database. And a half-changed database is one whose integrity has been lost.

Effects of an I/O Error

Suppose the program tries to read the stock-on-hand record and encounters a device error. Because of some I/O problem, the record cannot be obtained. Therefore, the program cannot run, and the customer order cannot be filled. In addition, any other work that depends on this record can no longer be completed.

Also, what if the program has already changed some records expecting that it could read and update this other record? Again, the overall integrity of the data has been lost, because it is only partially updated.

Effects of Queue Loss

In large systems, work requests are often saved (queued) and processed later, when the workload permits. Output that the programs send back to the requesters (or others) is also often queued and sent later, when the workload permits.

If the system fails between the time a request is entered (input) and the time it is taken off the queue and executed, the request might be lost. The same is true for queued output: If the system fails between the time the application program supplies its output and the time that output can be taken off the queue and actually sent, those results might be lost.

Requirements for a Recoverable System

In order for a system to be recoverable, it must protect the database from half-complete (and, therefore, incorrect) changes. It must be able to deal with the physical loss—unavailability or disappearance, in part or in whole—of the database. Also, it must protect the input and output queues from being lost.

The Mechanisms of Recovery in IMS

Successful recovery depends on two things:

- Having a safe point to return to, a known point of integrity. This is a point at which you know data is correct and make a record of it.
- Keeping track of what processing has been done since that safe point.

If these two types of information are gathered, recovery is almost always possible.

IMS is designed to be recoverable and has a number of mechanisms to help you in these areas. Some function automatically; others require your involvement.

Synchronization Points and Backup Copies

A synchronization point, or *sync point*, is a marker in time. It is a point in processing you can return to and start over from, a safe point. It is a known point of integrity, a safe place from which to rebuild.

There are two types of sync point. IMS itself creates periodic sync points (called *system checkpoints*), in which it records the current status of the system. In addition, application programs can cause sync points. These are points at which the work done by the application is assumed to be complete.

You can make backup copies of databases, and these copies can serve a function similar to the function of sync points.

Definition: An *image copy* is a backup copy of your data. Like a photograph, image copies are static and represent the state of the data at a particular instant in time. You can return to the image copy and restart processing.

Similarly, you can have IMS make backup copies of the message queue data sets, which contain the input and output messages. IMS system checkpoints usually do not include copies of the current state of the databases or queues.

Using Logs for Recovery

Logs are lists of activities, lists of work that have been done, and lists of changes that have been made. By knowing the state of the system when things were all right, and knowing what the system has done since then, you can recover it in the event of a failure.

In the online environment, IMS keeps a log called the online log data set (OLDS). The OLDS resides on a direct access storage device (DASD) and is later written to another data set called the system log data set (SLDS). The SLDS can reside on DASD, tape, or mass storage. In the batch environment, log records are written directly to an SLDS. IMS also keeps another log called the restart data set (RDS). The RDS contains information needed to restart IMS. IMS automatically handles the reading and writing of all of its logs.

Backout (Backward Recovery)

After you gather recovery information on the logs, there are two main ways of using it.

Definitions: *Forward recovery* involves reconstructing information and re-applying it to the database, and *backward recovery* (usually known as *backout*) involves removal of bad or incomplete information.

With forward recovery, you reconstruct information or work that has been lost, and add it to the database. With backout, you remove incorrect or unwanted changes from information. For an IMS subsystem, you perform the task of forward recovery (IMS does supply utilities to help). Backout of transactions (and their associated database updates) is handled automatically by IMS. Backout usually does not require your involvement, but an application program can control backout processing for its own transactions. Backout of batch programs can be done either automatically or using an IMS utility.

Recovery During Restart

IMS can recover from many types of failures during restart. In some cases, the recovery is automatic, for example, backout of transactions and the associated database updates. In other cases, you request a type of recovery on the restart command, for example, you can tell IMS to rebuild the message queues from a previous backup copy.

Recovery Utilities and Services

IMS has a number of utilities and services to help you recover and maintain a recoverable system:

- Database-related utilities: Use these utilities to make image copies of your databases, to accumulate and sort records of database changes, to speed up recovery, and to recover databases.

- System-related utilities: Use these utilities to help you manage the system by managing and recovering the log data sets or generating analysis reports.
- Transaction-Manager-related utilities: Use these utilities to help you manage and control your data communications and transactions.

Database Recovery Control

The IMS Database Recovery Control (DBRC) facility makes it easier for you to recover IMS databases by extending the capabilities of IMS utilities for database recovery. DBRC can help recover both DL/I databases and Fast Path data entry databases (DEDBs). DBRC offers two levels of control: log control and share control.

When you use log control only, DBRC controls the use and reuse of OLDSs for IMS.

When you use share control, DBRC:

- Controls the use and reuse of OLDSs for IMS
- Records recovery-related information in the Recovery Control (RECON) data set
- Assists in recovering databases
- Generates job control language (JCL) for recovery-related utilities
- Registers and controls the scheduling of databases

Automated Operations and Recovery

The IMS automated operator interface (AOI) allows application programs to issue IMS commands. It can also intercept IMS messages routed to the MTO. You can use it to develop procedures tailored to your installation and to automate some parts of the recovery process. This facility is described in Chapter 13, “Automated Operations,” on page 185.

In an IMS complex with the Extended Recovery Facility (XRF), AOI runs on the active IMS subsystem, and after an XRF takeover, AOI runs on the new active subsystem.

Recovery in an IMS DBCTL Environment

The IMS DBCTL environment allows one or more transaction management subsystems, such as CICS®, to access DL/I databases and DEDBs. These transaction management subsystems are known as *coordinator controllers* (CCTLs). A CCTL communicates with the DBCTL subsystem using an interface called the database resource adapter (DRA). Connections or paths between a DBCTL control region and a CCTL are created as part of CCTL initialization. These connections are called *threads*.

Because the CCTL is a separate subsystem, failures are isolated between the IMS DBCTL subsystem and the CCTL. That is, failure of the DBCTL subsystem does not generally cause the CCTL to terminate. Likewise, failure of a CCTL does not generally cause failure of the DBCTL subsystem. Generally, the only time a CCTL failure can affect the DBCTL subsystem is if the CCTL had one or more threads executing in DL/I at the time of its failure).

An IMS DBCTL subsystem cannot restart from log data not created by a DBCTL subsystem. Likewise, non-DBCTL IMS subsystems cannot restart using log data created by a DBCTL subsystem. In other words, restarts of an IMS DBCTL subsystem must use log data produced by that system. You can, however, perform database recovery using a combination of log data sets created by other subsystems: IMS DB/DC, batch, or DBCTL.

Because DBCTL requires the DBRC log control facility, you must use DBRC with DBCTL. Log records produced by an IMS DBCTL system are compatible with an IMS DB/DC system (except for subsystem restart).

Recovery in an XRF Complex

IMS XRF provides an alternate IMS subsystem that monitors the log data of the active IMS subsystem and takes over the processing load of the active subsystem if it experiences a failure. Everything about non-XRF recovery mechanisms is also true in an XRF complex, but with XRF, IMS can do more to reduce the time that data is unavailable to users after an abnormal event occurs.

You can specify conditions that trigger an automatic takeover, such as:

- A failure of the IMS control region
- A total failure of OS/390®
- A single central processor complex (CPC) failure
- An IRLM failure that requires an IMS STATUS exit routine
- A Virtual Telecommunications Access Method (VTAM®) failure that requires an IMS TPEND exit routine

You can also manually initiate an XRF takeover in a non-failure situation to introduce planned changes to your system with minimal disruption to users.

For a complete description of XRF, see Chapter 15, “Extended Recovery Facility (XRF),” on page 237.

Steps in the Process of Recovery

The process of recovery includes two major tasks:

- Planning—what you do before a problem occurs
- Reacting—what you do to fix a problem after it has occurred

The task of planning for IMS recovery can be further refined:

- Set up logging
- Establish checkpoints
- Make backup copies
- Create procedures and assign responsibilities

The task of reacting can also be further refined:

- Notice error
- Shut down IMS (only if necessary)—can be a full or partial shutdown
- Diagnose and fix (or possibly bypass) error
- Use recovery utilities and services (if necessary)—can be run online or offline
- Restart IMS (or the now fixed part of IMS)

Of course, real recovery is rarely as straightforward as these two lists suggest.

Complications in Recovery

If you could always recover a system simply by knowing where you started and keeping track of what you have done since then, it would be easy. Unfortunately, complications often arise, not only in the data and applications, but in the mechanisms of recovery themselves.

The IMS logs (on DASD or tape) are just as subject to I/O problems as any other data set. You can run out of space, or develop physical problems with the drive. The internal IMS modules involved in recovery are totally dependent on the operating system and hardware; if the system crashes for any reason (for example, power failure, abend, hardware malfunction), the recovery processing performed by these modules might be incomplete.

So, recovering from a problem can be a two-part process: first you must recover the IMS recovery tools themselves, then you can recover the data and applications.

What IMS Cannot Recover

IMS can automatically recover from some errors, and IMS provides mechanisms to allow you to recover from many more errors, but there are some errors IMS cannot recover. Generally, these unrecoverable errors are of two types:

- Application logic or input errors
- Operational errors, including misuse (accidental or malicious) of the IMS recovery facilities

Application Logic or Input Errors

When you detect an error caused by an application program that is running, backing out the faulty data is fairly simple. But if that program is given the wrong input or is constructed with logic errors, you might not be able to detect such problems because the program will run to completion. After the application program has finished running (or has committed its data), the faulty data can be used by other programs, and the problem can spread throughout your system.

IMS has no automatic or guaranteed way to back out committed data. You might be able to back out the data manually (for example, by editing individual records in the affected databases), but such a process is likely to be difficult. And the more time that passes after the original errors are introduced, the more programs are likely to have used the bad data, and the less likely your chances of success. Moreover, output that has been produced or actions taken as a result of the faulty data will already be outside the boundaries of the databases.

Recommendation: To minimize application logic or input errors, you should extensively test new applications (both individually and in an integrated system test) before bringing them online. Extensive validation of user input by application programs also helps minimize input errors.

Operational Errors

Successful recovery requires both proper operation of IMS on a daily basis and proper use of the recovery facilities. This proper use includes such things as:

- Using the correct data sets
- Avoiding improper job cancellations during regular operations
- Regularly maintaining the logs
- Using the correct log volumes in the correct sequence
- Using the appropriate utilities at the appropriate time

You should have no difficulties in operating IMS properly, and later sections in this book explain how to set up the necessary procedures and guidelines. However, if you operate IMS, its recovery facilities, or both, improperly, IMS might not be able to correct problems introduced by such improper use.

Overhead of Recovery

Although the value of recovery is obvious, it is not without cost. Logging, for example, takes time and space: some amount of processing time is required when a log record is created, and each log record uses space on a data set. Each checkpoint written also takes time and space, as does each backup copy made.

Having a basic recovery scheme is a practical necessity, not really a matter of choice. However, you do have a choice about how elaborate to make your recovery scheme.

Generally, the more extensive recovery preparations you build into your daily operations, the faster you can bring your system back into production after an error occurs. The trade-off is between speed of recovery and the daily overhead of recovery maintenance. You also need to consider the cost of having your online system down.

In judging this trade-off, consider how frequently errors occur that require recovery. Errors will certainly occur, but rare errors merit less recovery preparation than those that occur more frequently. You decision about the extent to which you prepare for recovery mechanisms should be based on knowing your installation’s needs and priorities, and weighing them against each other.

Chapter 2. Tools for IMS Operations and Recovery

IBM provides a number of tools for operations and recovery. The tools that are included with IMS are discussed in this chapter. They can be grouped into the following categories:

Logging
Synchronization Points
Backup
Shutdown of the IMS system
Restart
Backout
Automated Operations

Related Reading: Another very important tool that comes with IMS is Database Recovery Control (DBRC), which is described in *IMS Version 9: DBRC Guide and Reference*.

IBM also offers a number of IMS database productivity tools. IMS Support Tools is a set of database performance enhancements for your IMS environment. These tools can help you automate and speed up your IMS utility operations. They can also assist you in analyzing, managing, recovering, and repairing your IMS databases. You can learn more about these tools on the Web at http://www.ibm.com/software/data/ims/about/imstools/.

Logging

The log makes recovery and restart possible. As IMS operates, it constantly records its activities on the log. The log records:

- When IMS starts up and shuts down
- When programs start and terminate
- Changes made to the database
- Transaction requests received and responses sent
- Application program checkpoints
- System checkpoints

Recovery is possible because IMS records these activities on the log. Figure 1 on page 12 illustrates how logging works in an IMS online environment; logging in a batch environment is simpler.

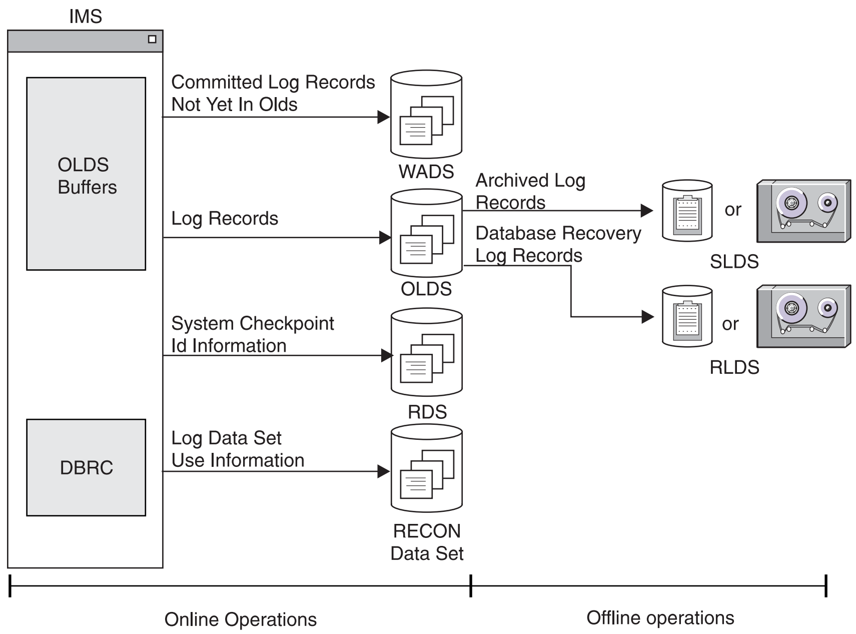

IMS
OLDS Buffers
DBRC
Committed Log Records Not Yet In Olds
Log Records
System Checkpoint Id Information
Log Data Set Use Information
WADS
OLDS
RDS
RECON Data Set
Archived Log Records
Database Recovery Log Records
or
SLDS
or
RLDS
Online Operations
Offline operations

Figure 1. Overview of the Logging Process

IMS externalizes log records by writing them to an Online Log Data Set (OLDS). To enhance performance and to optimize space on OLDSs, incomplete or partially filled buffers are written to a Write Ahead Data Set (WADS) when necessary for recoverability. The WADSs are high-speed DASD data sets with a high write rate. Only complete log buffers are written to OLDSs. When the log data is on an OLDS, the equivalent WADS records are ignored.

IMS uses a set of OLDSs in a cyclical way, which allows IMS to continue logging when an individual OLDS is filled. Also, if an I/O error occurs while writing to an OLDS, IMS can continue logging by isolating the defective OLDS and switching to another one. Once an OLDS has been used, it is available for archiving to a System Log Data Set (SLDS) on DASD or tape by the IMS Log Archive utility. The utility can be executed automatically through an IMS startup parameter (ARC=).

When IMS is close to filling the last available OLDS, it warns you so you can ensure that archiving completes for used OLDSs or add new OLDSs to the system. You can also manually archive the OLDSs to SLDSs using the IMS Log Archive utility. An SLDS can reside on DASD or tape. After an OLDS is archived, it can be reused for new log data. You use SLDSs as input to the database recovery process.

When you archive an OLDS, you can request that IMS write a subset of the log records from the OLDS to another log data set called the recovery log data set (RLDS). An RLDS contains only those log records required for database recovery.

While IMS is running and logging its activities, it takes periodic system checkpoints, and writes the checkpoint notification to another data set called the restart data set (RDS). IMS uses the RDS when it restarts to determine the correct checkpoint from which to restart.

In a batch environment, IMS writes log records directly to an SLDS, and does not use either the OLDSs or WADSs. Just as in the online environment, batch SLDSs can reside on DASD, tape, or mass storage. If the SLDS is on DASD, you can use an IMS utility to copy log records from DASD either to tape or to other DASD.

Synchronization Points

Strictly speaking, all you need to be able to restart a system after a failure is the log: as long as you start out correctly and log everything you do thereafter, you can always recover. Restart in this case would require the system to read every log record ever written for the system. The more time that has passed since the first time you started the system, the more log records will have been written, and the more time-consuming the restart will be.

Definition: A synchronization point, or *sync point*, is a known point of data or system integrity. It is a point in time at which all of the changes made to the database are complete. If a failure makes restart necessary, you can begin reprocessing from the most recent sync point.

The value of having a sync point is that when IMS restarts, it can ignore all logs created before the sync point was taken. If your sync points are infrequent, they become less valuable because the amount of log data IMS has to read during a restart increases. Thus, you should establish periodic sync points. Then, if a problem arises, you do not need to examine as much of the log, because you established the last sync point relatively recently.

IMS uses two types of sync points: those taken by IMS itself (called *system checkpoints*), and those taken by individual application programs running under IMS (called application program *sync points* or application program *commit points*).

System Checkpoints

IMS automatically takes periodic system checkpoints, so if it is necessary to restart IMS, it can begin at the last checkpoint (or an earlier one if the work done before the checkpoint was not complete).

The interval between system checkpoints is measured by how many log records IMS writes, so the checkpoints are a measure of activity rather than elapsed time. You select the interval between system checkpoints when you install IMS.

As with most choices involving preparation for recovery, the overhead of taking frequent checkpoints must be weighed against the advantages of faster recovery. Because IMS does not suspend work to take a checkpoint, decreased transaction speed is not a significant overhead factor.

Recommendation: Do not suppress system checkpoints to improve system performance because emergency restart must always go back at least two system checkpoints (or back to a prior restart).

The IMS MTO can also request an additional system checkpoint at any time by issuing a `/CHECKPOINT` command.

IMS takes a system checkpoint when any of the following occur:

- IMS starts
- A regular interval has elapsed
- The MTO enters a command (`/DBRECOVERY`, `UPDATE DB STOP(ACCESS) OPTION(FEOV)`, `/CHECKPOINT`, or `/SWITCH OLDS CHECKPOINT`)

- IMS shuts down

Application Sync Points

IMS system checkpoints help you recover the IMS subsystem, but you also need to be able to restart application programs if they fail. IMS allows application programs, both batch and online, to take sync points.

An application program sync point has two purposes:

- It marks an intermediate completion point—a place at which the work finished so far is judged to be correct. Any future recovery can take place from this point.
- It frees locks held on database records the program has updated and enqueues any output messages the program created. IMS can send enqueued messages to their destinations.

Application program sync points are sometimes called *commit points*. By taking a sync point, the program is committing itself to the accuracy and completeness of what it has done, and releasing the data for use by other applications.

Backup

A *backup copy* of a data set serves the same purpose for a recoverable system as a checkpoint: it defines a place from which you can restart processing. IMS provides utilities to allow you to make several types of backup copies:

- Database backup copies
- Message queue backup copies
- System data set backup copies

Database Backup Copies

When IMS takes a regular system checkpoint, it records internal control information for DL/I and for Fast Path, but it does not record any of the contents of the database. If the database is lost, examining the last system checkpoint does not allow you to recover it. The system log can tell you what changes have occurred, but without the original database itself, recovery can be impossible.

Recommendation: Make a backup copy of all databases after you initially load them. You should also make new backup copies at regular intervals. More recent backup copies require fewer log change records to be processed during recovery, and thus the time needed for recovery is reduced.

IMS provides several utilities for making backup copies of a database:

- Image copy utilities

  The IMS database image copy utilities allow you to take a “snapshot” of a database before and after changes have been to the database. These snapshots are called *image copies*.

  Definition: An image copy is an as-is image of a database. The image copy utilities do not alter the physical format of the database as they copy it. Image copies are backup copies of your data that help speed up the processes of database recovery and backout.

  The image copy utilities are Database Image Copy utility (DFSUDMP0), Database Image Copy 2 utility (DFSUDMT0), and Online Database Image Copy utility (DFSUICP0).
- The HISAM Reorganization Unload utility (DFSURUL0)

This utility allows you to process an entire HISAM database in one pass (the image copy utilities process each database data set individually) while it is unloaded and reorganized. This utility runs while the database is offline.

After you unload the database, you use the HISAM Reorganization Reload utility (DFSURRL0) before bringing the database back online. If you do not reload the database, an application program can use the database, but will use the old organization of the data set. Your backup copy will not match the log records produced for the database, and you will not be able to use the backup copy to recover the database without damaging your data integrity.

You can run these utilities with or without DBRC. You can also use various MVS utilities to make your backup copies, but these utilities do not interact with DBRC, which could cause integrity problems depending on how your IMS system is defined.

Message Queue Backup Copies

Messages (the transaction requests entered by end users, and the responses going back to them) are stored on queues before being processed. If you are not sharing the IMS message queues, messages are stored in the message queue data set, which resides partly on disk and partly in virtual storage. In a shared-queues environment, IMS messages are kept on a coupling facility. When IMS takes a regular system checkpoint, it does not record the contents of the message queues, just as it does not record the contents of the databases.

When you shut down IMS normally (rather than abnormally), any changed messages in virtual storage (not on a coupling facility) are automatically written to the disk portion of the message queue data set. Therefore, if you periodically shut down IMS, there is little or no need to backup the message queues manually.

When you run IMS for extended periods without shutting it down, you might want to back up the message queues periodically. In a non-shared-queues environment, use the `/CHECKPOINT SNAPQ` command to back up the message queues; this command does not shut down IMS.

In a shared-queues environment, you must periodically back up the shared queues. Use the `/CQCHKPT` command to copy IMS messages to the Common Queue Server’s structure recovery data set.

Backing up the message queues reduces the time required for recovery if problems arise with the message queue data sets.

System Data Set Backup Copies

In addition to making database backup copies and message queue backup copies, you should also make backup copies of the IMS system data sets. These include the ACB library (IMS.ACBLIBx) and the MFS library (IMS.FORMATx).

IMS does not provide any special utilities or commands to make backup copies of system data sets. You can, however, make periodic backup copies using MVS utilities, such as IEBCOPY. You can make these copies at the same time you make periodic backup copies of MVS system libraries. You may also find it convenient to back up the system data sets and databases at the same time.

When you make online changes, you should make backup copies of the active data sets after you switch the inactive and active data sets.

You should also periodically back up RECON data sets using the `BACKUP.RECON` command.

Shutdown of the IMS System

Certain problems are severe enough that they cause the IMS subsystem to fail. In these cases, IMS shuts itself down. In other cases, however, IMS keeps running, so you must shut it down manually before you can perform recovery. You can shutdown either part of the IMS subsystem (partial shutdown), or you can shut down all of it (full shutdown).

Recommendation: Because a primary goal is to keep IMS function available to users, you should only use a full shutdown when a partial shutdown does not recover the system.

When only a part of IMS is malfunctioning, you might be able to shut down only that part and leave the rest of IMS functioning productively. For example, a faulty database can be taken offline (made unavailable to application programs), while IMS and the other databases continue processing. Or a terminal that is malfunctioning can be detached, while all other functions continue unaffected.

To shut down only part of IMS, use a command that stops the component that is malfunctioning.

You can also shut down IMS in a controlled manner. A controlled shutdown is desirable because it saves current information in the system, and allows an easy and accurate restart of the system at a later time.

Shut down IMS using the `/CHECKPOINT` command with one of the following keywords: `FREEZE`, `DUMPQ`, or `PURGE`.

Restart

Generally, the way you restart IMS corresponds to the way you shut it down, or the way it fails.

- If you only shut down part of IMS (for example, take a database offline or detach a line), you need to restart only that part.
- If you shut down all of IMS or if IMS fails, you need to restart the whole IMS subsystem.

After recovering a failed component, you can restart it. For example, if you take a database offline because of an I/O error and then recover it, you can again make it available to applications.

To restart only part of IMS, use a command that starts the component that is stopped.

Before you can restart the entire IMS subsystem, you must first perform all necessary recovery. You can then restart IMS in one of three ways:

- *Normal restart* initializes an IMS subsystem that has not failed. A normal restart can be either a *cold start* which restarts IMS without reference to any previous execution of IMS, or a *warm start* which restarts an IMS subsystem that was terminated using a `/CHECKPOINT` command. The normal restart command is `/NRESTART`.

- *Emergency restart* initializes an IMS subsystem that has failed. During an emergency restart, IMS resets transactions and active regions, and restores databases and message queues to the most recent sync point. You must manually restart batch and batch message processing (BMP) regions. The emergency restart command is `/ERESTART`.
- *Automatic restart* reduces MTO intervention and can make restart faster because IMS automatically chooses the appropriate restart command. The operator does not enter a restart command; instead you specify automatic restart by coding `AUTO=Y` in the JCL for the IMS control region.

Backout

When IMS or an application program fails, you need to remove incorrect or unwanted changes from the database. Backward recovery or *backout* allows you to remove these incorrect updates. The three types of backout are:

1. Dynamic backout
2. Backout during emergency restart
3. Batch backout

IMS performs the first two types of backout automatically, and you initiate the last one manually.

IMS automatically (dynamically) backs out database changes in an online environment when any of the following occurs:

- An application program terminates abnormally
- An application program issues a rollback call
- An application program tries to access an unavailable database
- A deadlock occurs

In a batch environment, you can specify that IMS should dynamically back out database changes if the batch job abends, or issues a rollback call.

During restart processing after a system failure, IMS determines if any application programs were executing at the time of failure and if they made changes that need to be backed out. Before IMS restarts, it scans log records for these changes, and then backs out the changes from the affected databases. If IMS runs out of memory while scanning the log, you could get a message telling you to back out the changes manually.

You can use the IMS Batch Backout utility (DFSBBO00) to remove database changes made during execution of a DL/I or DBB region or an online program. You can use the utility to back out changes since the last checkpoint. You can select a specific checkpoint if the batch region does not use data sharing, and if it is not a BMP. For BMPs, do not specify a checkpoint because the utility always backs out BMPs to the last checkpoint on the log.

If dynamic backout or backout during emergency restart fails, IMS stops the databases for which backout did not complete, and retries the backouts when you restart the databases.

Automated Operations

Automated operations are tools and techniques that help you improve your installation's productivity. As your system grows more complex and your message traffic increases, automating certain tasks can increase your system's efficiency. You can use automated operations to:

- Minimize errors
- Increase availability
- Expedite problem diagnosis and prevention

IMS provides the following tools to help you automate your operations:

- Time-Controlled Operations (TCO)
- Automated Operator Interface (AOI)
- REXX SPOC API

IMS operations can also be automated using NetView®. Because NetView functions independently of IMS, it can gather information and issue commands that are not available to IMS.

Related Reading: For more information about NetView, see *Tivoli NetView for z/OS Installation: Getting Started* or *Tivoli NetView for z/OS User's Guide*.

Advantages of Automation

Many of the jobs that operators perform are simple, repetitive tasks, such as monitoring the system and issuing recovery commands. You can often automate these jobs and thereby free the operator for more complex activities, such as implementing operational procedures.

In the IMS environment, TCO and AOI can improve your operator productivity by:

- Improving operator productivity and accuracy. Automating operator tasks simplifies procedures, reduces operator input, and minimizes operator errors. For example, TCO can reduce operator input by automatically monitoring system status, starting message regions and telecommunication lines, and notifying users of system status.
- Expediting problem determination. An operator's primary job is to bypass or fix problems quickly. If adequate information about a problem is not available, it may not be possible for the operator to fix it quickly. Automated operations are especially suited for problem determination. For example, TCO can automatically gather necessary information and do diagnostic analysis so that a problem can be quickly identified and corrected.

Deciding What to Automate

Identifying operations that can be automated and implementing them requires a sound understanding of your installation's operations. You must collect and analyze various resources to identify which tasks are good candidates for automation. Resources you should analyze include:

- System logs
- Problem management reports
- Record of calls to the help desk
- Operators' notes

Repetitive and predictable tasks are good candidates for automation.

The process of automating operations should be an iterative one. After you have developed and used automated procedures, you should evaluate them and, in the process, consider whether any new tasks can be automated.

Part 2. Operating an IMS Subsystem

Chapter 3. Commands for IMS Tasks

Table 2 lists IMS tasks and the operator commands that you issue to accomplish these tasks. For detail on the commands themselves, see *IMS Version 9: Command Reference*.

Related Reading: See "List of Commands with Similar Functions for Multiple Resources" on page 105 for a list of commands that perform similar functions for different resources.

Table 2. IMS Tasks and Commands

Task	Command
Abending IMS	F job,STOP F job,STOPxxx F job,FORCExxx
Abending IMS with a dump	F job,DUMP F job,DUMPxxx
Abending IRLM	F irlmproc,ABEND
Aborting global online change on all IMSs in an IMSplex	TERMINATE OLC
Aborting online reorganization in an OM environment	TERMINATE OLREORG
Aborting online reorganization in a non-OM environment	/TERMINATE OLREORG
Activating VTAM nodes	/ACTIVATE
Activating MSC links	/ACTIVATE
Adding information to the RECON data set	/RMNOTIFY
Allocating a conversation with an LU name and TP name	/ALLOCATE
Altering assignments for IMS resources	/MSASSIGN
Altering relationships between IMS resources	/ASSIGN
Assigning a user structure to a nonexistent node	/OPNDST NODE x USER y
Cancelling an input message	/CANCEL
Cancelling an output message	/DEQUEUE
Changing IMS resources	/CHANGE UPDATE
Changing IMS Fast DB Recovery surveillance	/CHANGE
Changing online reorganization RATE or data set disposition in an OM environment	UPDATE OLREORG
Changing online reorganization RATE or data set disposition in a non-OM environment	/UPDATE OLREORG
Changing the RECON data set	/RMCHANGE
Changing the ready state for a terminal component	/COMPT /RCOMPT

Table 2. IMS Tasks and Commands (continued)

Task	Command
Checkpoint, taking	/CHECKPOINT
Class, assigning	/ASSIGN UPDATE TRAN SET(CLASS)
Cold start of components of IMS	/ERESTART
Cold start of IMS	/NRESTART
Connecting to a restarted IRLM	F job,RECONNECT
Coupling facility structures, listing status	/CQQUERY
CQS checkpoint, initiating	/CQCHKPT
CQS structure checkpoint, requesting	/CQSET
Creating records in the RECON data set	/RMINIT
DBRC, commands for	/RMxxxxxx
Deallocating a user for an ISC node	/QUIESCE
Delivering asynchronous output from an LU 6.2 device	/ALLOCATE
Deleting affinities from VTAM	/CHECKPOINT
Deleting IMS resources	/CHANGE /CHECKPOINT
Deleting information in the RECON data set	/RMDELETE
Deleting security	/DELETE
Destination of messages, setting	/SET
Directing output to a component of a PTERM	/ASSIGN
Disconnecting a VTAM terminal	/CLSDST /RCLSDST
Displaying IMS Fast DB Recovery status	F fdbrproc,STATUS
Displaying IMS resources and status	/DISPLAY QUERY STATUS
Displaying IMSplex member status or attribute information	QUERY MEMBER
Displaying IRLM status	F irmlproc,STATUS ,ALLD ,ALLI ,MAINT ,STOR ,TRACE
Displaying the LTERM name of the MTO	/RDISPLAY
Displaying members of the IMSplex	QUERY IMSPLEX
Displaying OLCSTAT data set information	QUERY OLC LIBRARY (OLCSTAT)
Displaying status of a database or area	/DISPLAY DB /DISPLAY AREA QUERY DB QUERY AREA
Displaying status of online reorganization	QUERY OLREORG
Dumping MSDBs	/DBDUMP UPDATE DB STOP(UPDATES)

Table 2. IMS Tasks and Commands (continued)

Task	Command
Ending special modes	/END
Exclusive mode for a terminal, setting	/EXCLUSIVE
External subsystem, entering commands for	/SSR
Fast DB Recovery surveillance, changing	/CHANGE
Formatting a terminal screen	/FORMAT
Generating JCL for utilities	/RMGENJCL
IMS monitor, starting or stopping	/TRACE
Initiating IMS Fast DB Recovery of tracked databases	F fdbrproc,RECOVER
Initiating a session with a terminal	/OPNDST
Initiating an online reorganization in an OM environment	INITIATE OLREORG
Initiating an online reorganization in a non-OM environment	/INITIATE OLREORG
IRLM, abending	F irlmproc,ABEND
IRLM, changing MAXCSA	F irmlproc,SET,CSA=nnn ,TRACE=nnn
IRLM, displaying status	F irmlproc,STATUS ,ALLD ,ALLI ,MAINT ,STOR ,TRACE
IRLM, reconnecting to	F proc, RECONNECT
IRLM, releasing locks	F irlmproc,PURGE,imsname
IRLM, setting timeout value	F irlmproc,START,TMOUT=n F irlmproc,STOP,TMOUT
IRLM, starting	S irmlproc
IRLM, stopping	P irmlproc
IRLM, tracing	S irmlproc,TRACE=YES TRACE CT
ISC node shutdown	/QUIESCE
ISC session cold start	/ASSIGN
Language Environment® (LE) attributes and parameters, changing	UPDATE LE DELETE LE
Limit count for a transaction, setting	/ASSIGN UPDATE TRAN SET(LCT)
Listing information in the RECON data set	/RMLIST
Logging off from a terminal	/RCLSDST
Logging on to IMS from a VTAM terminal	/OPNDST
LTERM to node, assigning	/ASSIGN
LTERM to PTERM, assigning	/ASSIGN
LTERM to user or ISC half session, assigning	/ASSIGN

Table 2. IMS Tasks and Commands (continued)

Task	Command
Message control error exit routine, activating	/DEQUEUE
Modifying IMS resources	/MODIFY
Modifying IMS resources globally for an IMSplex	INITIATE OLC
MTO, displaying LTERM name, line, and node	/RDISPLAY
MFS format, using on a terminal	/FORMAT
MFS test mode for terminals, setting	/TEST
Online change, performing	INITIATE /MODIFY
Output errors, testing for	/LOOPTEST
Output segments for application program, assigning	/ASSIGN
Output to secondary MTO, sending	/SMCOPY

Preventing updates to a database	/DBDUMP
	/DBRECOVERY
	UPDATE AREA STOP(ACCESS)
	UPDATE DATAGRP STOP(ACCESS)
	UPDATE DB STOP(UPDATES)
	UPDATE DB STOP(ACCESS)

Priority for a transaction, assigning	/ASSIGN UPDATE TRAN SET(NPRI)
RACF®, controlling	/SECURE
Reconnecting to a restarted IRLM	F job,RECONNECT
Reconnecting to coupling facility structures	F job,RECONNSTR
Recovering databases using the database recovery service	/RECOVER
Releasing IRLM locks	F irlmproc,PURGE,imsname
Remote takeover, initiating	/RTAKEOVER
Removing IMS from a VTAM generic resources group	/CHECKPOINT /STOP
Removing an IMS from the IMSplex	/CHE FREEZE or DUMPQ or PURGE LEAVEPLEX
Resetting values set during IMS system definition	/ASSIGN UPDATE

Restarting a component of IMS	/OPNDST
	/RELEASE
	/RSTART
	/START
	/UNLOCK
	UPDATE TRAN START(Q,SCHD,TRACE)
	UPDATE AREA START(ACCESS)
	UPDATE DATAGRP START(ACCESS)
	UPDATE DB START(ACCESS)
	UPDATE DB SET(LOCK(OFF))

Restarting IMS after failure	/ERESTART
Restarting XRF alternate subsystem	/ERESTART

Table 2. IMS Tasks and Commands (continued)

Task	Command
Resuming a conversation	/RELEASE
Scheduling a suspended transaction	/DEQUEUE UPDATE TRAN START(SUSPEND)
Secondary MTO, sending output to	/SMCOPY
Security, controlling	/SECURE
Security, deleting	/DELETE
Sending messages to terminals or other IMS subsystems	/BROADCAST
Setting timeout value for IRLM	F irlmproc,START,TMOUT=n F irlmproc,STOP,TMOUT
Sharing startup JCL between IMS dependent regions	/START
Shutting down CQS	P cqsproc
Shutting down the Common Service Layer (CSL)	F sciproc, SHUTDOWN CSLLCL/CSLPLEX (CSLLCL or CSLPLEX is required)
Shutting down IMS	/CHECKPOINT
Shutting down an ISC node	/QUIESCE
Shutting down OM	P omjob
Shutting down RM	P rmjob
Shutting down SCI	P scijob
Signing on to IMS from a non-VTAM terminal	/IAM
Signing on or off IMS terminals	/SIGN
Starting CQS	S cqsproc
Starting IMS	/NRESTART /ERESTART
Starting IMS Fast DB Recovery	S fdbrproc

Starting IMS resources	/START
	UPDATE TRAN START(Q,SCHD,TRACE)
	UPDATE AREA START(ACCESS)
	UPDATE DATAGRP START(ACCESS)
	UPDATE DB START(ACCESS)

Starting IRLM	S irmlproc
Starting a line, link, node, PTERM, or user	/RSTART
Statistics, recording to IMS log	/CHECKPOINT

Stopping database access	/LOCK
	/STOP
	UPDATE AREA STOP(SCHD)
	UPDATE DATAGRP STOP(SCHD)
	UPDATE DB SET(LOCK(ON))
	UPDATE DB STOP(SCHD)

Stopping IMS resources	/STOP
	UPDATE AREA STOP(SCHD)
	UPDATE DATAGRP STOP(SCHD)
	UPDATE DB STOP(SCHD)
	UPDATE TRAN STOP(Q,SCHD)

Table 2. IMS Tasks and Commands (continued)

Task	Command
Stopping input to a line, link, or terminal	/PURGE
Stopping input messages with a transaction code	/PURGE UPDATE TRAN START(SCHD)STOP(Q)
Stopping IMS Fast DB Recovery	F fdbrproc,DUMP F fdbrproc,STOP
Stopping IMS Fast DB Recovery tracking	F fdbrproc,TERM
Stopping IRLM	P irmlproc
Stopping an LTERM, node, program, PTERM, or transaction	/LOCK SET(LOCK(ON))
Stopping a transaction	/STOP UPDATE TRAN STOP(Q,SCHD)
Stopping output to a line, link, or terminal	/PSTOP
Stopping output to a programmable remote station	/MONITOR
Stopping scheduling of a message with a transaction code	/PSTOP UPDATE TRAN STOP(SCHD) START(Q)
Subsystem, external, entering commands for	/SSR
Suspending a conversation	/HOLD
Switching data sets for XRF	/SWITCH
Terminating I/O for terminals	/IDLE
Terminating preset mode	/RESET
Terminating special modes	/END
Terminating a conversation	/EXIT
Test mode for terminals, setting	/TEST
Testing for output errors	/LOOPTEST
Tracing IMS resources	/TRACE UPDATE TRAN START(TRACE)
Tracing IRLM	S irmlproc,TRACE=YES TRACE CT
Transport Manager Subsystem, defining	Define
Transport Manager Subsystem, displaying	Display
Transport Manager Subsystem, setting	Set
Transport Manager Subsystem, starting	Start
Transport Manager Subsystem, stopping	Stop
Undoing /ASSIGN	/ASSIGN /NRE CHECKPOINT 0 UPDATE TRAN
Undoing /EXCLUSIVE	/END /START
Undoing /HOLD	/RELEASE /EXIT
Undoing /IAM	/IAM
Undoing /IDLE	/ACTIVATE

Table 2. IMS Tasks and Commands (continued)

Task	Command
Undoing /LOCK	/UNLOCK UPDATE DB SET(LOCK(OFF))
Undoing /LOOPTEST	/END
Undoing /MSASSIGN	/MSASSIGN /NRE CHECKPOINT 0
Undoing /SET	/RESET
Undoing /START, /RSTART, /PSTART, /PURGE, or /MONITOR	/STOP
Undoing /START, /STOP, /COMPT, /RCOMPT, /PSTART, /PURGE, or /MONITOR	/RSTART
Undoing /START, /MONITOR, /RSTART, /STOP, or /PSTART	/PURGE
Undoing /START, /MONITOR, /RSTART, /STOP, or /PURGE	/PSTOP
Undoing /START, /RSTART, /STOP, /PSTOP, or /PURGE	/MONITOR
Undoing /TEST	/END /IAM /START
Unloading an area from a data space	/VUNLOAD
Verifying local transactions and LTERMs with remote transactions and LTERMs	/MSVERIFY
Verifying /MSASSIGN	/MSVERIFY
Warm start of IMS	/NRESTART
Writing a message to the IMS log	/LOG
XRF, switching data sets	/SWITCH

Important: In an IMSplex, type-2 commands are issued to the Operations Manager, which then routes the commands to the subsystem IMSplex members. When commands are issued globally in an IMSplex, they can behave differently from commands that are issued to single IMSs. For more information on commands in an IMSplex, see the *IMS Version 9: Common Service Layer Guide and Reference* and the *IMS Version 9: Command Reference*.

Restrictions: Some restrictions apply to IMSs running HALDB Online Reorganization (OLR):

- The following commands cannot be processed against a HALDB partition on an IMS system while OLR is running against that partition on the same IMS system:
  - `/START DB`
  - `/DBRECOVERY DB`
  - `/DBDUMP DB`
  - `/STOP DB`
  - `UPDATE DB START(ACCESS)`
  - `UPDATE DB STOP(ACCESS|SCHD|UPDATES)`

- The following commands cannot be issued against a HALDB master while OLR is reorganizing any of its partitions:
  - `/DBRECOVERY DB`
  - `/DBDUMP DB`
  - `/START DB` with `ACCESS=UP`
  - `UPDATE DB START(ACCESS) SET(ACCTYPE(UPD))`
  - `UPDATE DB STOP(ACCESS|UPDATES)`

Related Reading: For more information about these commands, see *IMS Version 9: Command Reference*. For more information about HALDB OLR support, see *IMS Version 9: HALDB Online Reorganization Guide and Reference*.

Chapter 4. Using IMS Application Program Sync Points

As described in “Synchronization Points” on page 13, IMS application programs can (and should) take checkpoints. This explains how these checkpoints and system sync points affect IMS operations.

The Commit Process

During an application’s sync-point processing, IMS creates a log record to establish commitment of database changes and availability of output messages. The commit process is not complete until IMS physically writes this log record to the OLDS because an incomplete set of database change and message records exist on the log for system restart.

The commit processes work differently for DL/I and Fast Path applications. For DL/I, IMS makes database changes in the buffer pool at the time of a DL/I call, and may or may not write them to disk before the commit point. If you restart the system, IMS backs out these uncommitted changes by using the log. IMS stores inserted message segments in the message queue and must similarly discard them.

For Fast Path, the approach is different. IMS keeps all changes in storage until it physically logs the commit record. Only then does IMS write database changes to DASD and send output messages. Because no changes appear on external storage (except for the log) until the commit record is written, IMS does not need to perform backout processing for the database—IMS only needs to discard the updates in storage. With Fast Path, system restart ensures that IMS writes committed updates to DASD and sends output messages.

Relationship Between Checkpoints and Sync Points

IMS keeps track of all checkpoints and sync points. IMS usually uses a sync point during recovery, but returns to the checkpoint in the following situations:

- For a full recovery in the DB/DC environment, IMS returns to the earliest of either the checkpoint prior to the current checkpoint or the checkpoint prior to the first uncommitted application program update.
- For a full recovery in the DBCTL environment, IMS always returns to the checkpoint before the first uncommitted application program update.
- For a full recovery in the DCCTL environment, IMS always returns to the checkpoint before the latest system checkpoint.
- In the DB/DC or DCCTL environments, if a BUILDQ is requested on the restart, IMS returns to the last SNAPQ or DUMPQ checkpoint. IMS returns to this checkpoint even if it is older than the checkpoint normally needed for the restart.

In Figure 2 on page 38, for example, if a system-wide failure occurs in the DB/DC environment just after the MTO takes a system checkpoint but just before program Beta commits (assuming program Alpha has not made any updates since its last commit), IMS must return to the system checkpoint before Beta started.

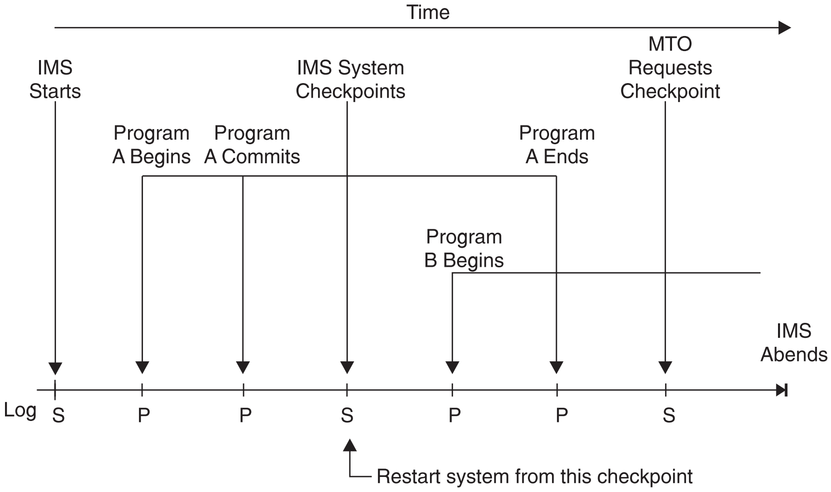


Figure 2. Independence of System Checkpoints and Application Sync Points

Synchronization Point Processing in CPI Communications-Driven Programs

For CPI Communications-driven programs running under Advanced Program-to-Program Communications for IMS (APPC/IMS), the application programs control their own sync-point processing. An application program can issue certain CPI Resource Recovery calls: `SRRCMIT` calls to commit data and `SRRBACK` calls to back out data. The IMS-managed (local) protected resources include:

- IMS TM message-queue messages
- IMS DB databases
- DB2 UDB for z/OS® databases

The highest level of synchronization supported for a conversation is SYNCPT, so CPI Communications-driven applications can have protected conversations.

IMS can be either the sync-point manager or the resource manager, depending on the setting of the sync-point level. For SYNC_LEVEL=NONE or CONFIRM, IMS is the sync-point manager, but for SYNC_LEVEL=SYNCPT, RRS/MVS is the sync-point manager and IMS is the resource manager.

Two-Phase Commit in the Synchronization Process

Application programs in a DBCTL, DCCTL, DB/DC, APPC/IMS, or OTMA environment can be involved in a two-phase commit process to record a sync point. At the completion of a two-phase commit, the resource manager commits database and message changes. The two phases are:

1. Phase 1, in which the sync-point coordinator directs sync point preparation and asks the connected resource managers whether updates to connected databases can be committed.

   The sync-point coordinator can be:

   - An IMS DB/DC subsystem for its resource managers and attached databases.
   - An IMS DCCTL subsystem for attached databases.

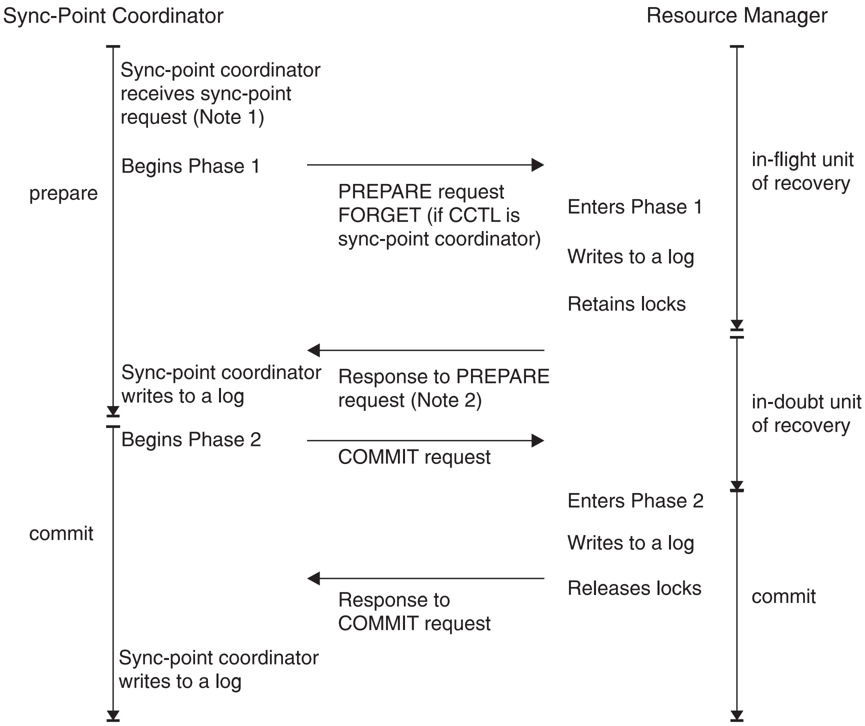
- A Coordinator Controller (CCTL) subsystem for units of work associated with the CCTL region. IMS DB acts as a resource manager when connected to a CCTL and also when accessed by ODBA application programs via the Open Database Access (ODBA) interface.
- OS/390 Resource Recovery Services/MVS (RRS/MVS) for its protected conversations with APPC/IMS applications programs or OTMA clients. IMS acts as a resource manager when connected to RRS/MVS.

2. Phase 2, in which the sync-point coordinator directs commit or abort processing and states that the resources must either be committed or aborted.

   In the DBCTL environment, if an application program makes no update DL/I calls or makes only inquiry-type DL/I calls, the CCTL requests a “forget” response to Phase 1 (if forget processing has been enabled). This means that only a limited Phase 2 occurs for that application program because no database resources have been altered. See *IMS Version 9: Customization Guide* for details on how to enable forget processing.

   The sync-point coordinator can request an abort without a Phase 1.

Figure 3 shows the two phases of the sync-point cycle for an IMS DBCTL environment and describes the activities taking place.


Sync-Point Coordinator
Resource Manager
Sync-point coordinator receives sync-point request (Note 1)
Begins Phase 1
prepare
PREPARE request FORGET (if CCTL is sync-point coordinator)
Enters Phase 1
Writes to a log
Retains locks
in-flight unit of recovery
Sync-point coordinator writes to a log
Response to PREPARE request (Note 2)
in-doubt unit of recovery
Begins Phase 2
COMMIT request
Enters Phase 2
commit
Writes to a log
Releases locks
Response to COMMIT request
commit
Sync-point coordinator writes to a log

Notes:

1. If the resource manager indicates that it cannot commit the updates, the sync-point coordinator should abort the unit of recovery, and the rest of this figure does not apply.
2. If the sync-point coordinator tells the resource manager to commit the updates, then it **must** commit.

Figure 3. Two-Phase Commit Process

Unit of Recovery

A *unit of recovery* (UOR) is the work done by a thread (connection between a resource-manager control region and a sync-point coordinator) during a sync-point interval, that is between two sync points.

In-Flight Unit of Recovery

The unit of recovery is said to be *in-flight* from its creation or its last sync point until the resource manager logs the end of Phase 1. If a resource manager fails before or during Phase 1 and is subsequently restarted, IMS aborts all database updates.

In-Doubt Unit of Recovery for DBCTL Connected to CCTL

From the time that the resource manager issues its response to the PREPARE request (the completion of Phase 1), to the time it receives a COMMIT or ABORT request from the CCTL, units of recovery are said to be *in-doubt*. When the resource manager is restarted after a failure, it tells the CCTL which in-doubt UORs exist, if any. The CCTL then takes action to resolve these in-doubt UORs. This is called resolve in-doubt processing, or resynchronization. If a CCTL cannot resolve all in-doubt UORs, you can use IMS or CCTL commands to display the units of recovery and take appropriate actions for committing or aborting them.

Recovery Tokens for DBCTL Connected to CCTL

A *recovery token* is a 16-byte identifier for each unit of recovery. The resource manager validates the recovery token to protect against duplication of units of recovery. In the DBCTL environment, you can display the recovery token using the IMS `/DISPLAY CCTL` command. The recovery token is the primary identifier used by DBRC, which performs unit-of-recovery management. DBRC keeps track of backouts that are appropriate for the Batch Backout utility to perform.

Recoverable In-Doubt Structure: An IMS DBCTL subsystem builds a recoverable in-doubt structure (RIS) for each in-doubt UOR when any of the following occurs:

- A CCTL fails
- A CCTL thread fails
- A resource manager fails

The resource manager uses a recoverable in-doubt structure during reconnecting to the CCTL if in-doubt UORs existed when either the CCTL or the resource manager failed. IMS logs all recoverable in-doubt structures during system checkpoints.

A recoverable in-doubt structure contains the following information:

- The recovery token in a residual recovery element (RRE)
- Changed data records in an in-doubt extended error queue element (IEEQE)
- An indication of data that is inaccessible because of unresolved in-doubt UORs
- Links to other recoverable in-doubt structures using extended error queue element (EEQE) queue elements (EQELs)

DBCTL Single-Phase Commit

A CCTL communicating with just one resource manager (IMS DBCTL subsystem) can request a sync point using just a single phase. If the CCTL communicates with more than one resource manager, it must use the two-phase commit process.

When the CCTL decides to commit a UOR, it can request a single-phase sync point. Single-phase commit can affect the recoverability of in-doubt data. A transaction is only in-doubt for the short time between the sync-point request and

DBCTL's commit. IMS can recover in-doubt data after a thread failure during single-phase commit, but cannot recover in-doubt data after a subsystem failure.

Sync-Point Log Records

During the two-phase commit process, IMS creates log records to establish the commitment of database changes (see “The Commit Process” on page 37). All of these log records can be used by the IMS Change Accumulation and recovery utilities.

All online log records involving the sync-point cycle contain a recovery token. This token ensures that IMS can recover and restart each unit of recovery. The sequence of log records for a unit of recovery reveals the sync-point cycle that it followed.

IMS logs the following records during the sync-point process:

X'08'	Schedule record
X'07'	Unschedule (terminate) record
X'0A08'	CPI Communications-driven application program schedule record
X'0A07'	CPI Communications-driven application program unschedule (terminate) record
X'5937'	Fast Path start commit
X'5938'	Fast Path start abort
X'5610'	Start of Phase 1
X'5611'	End of Phase 1
X'3730'	Start of Phase 2 Commit
X'5612'	End of Phase 2 Commit
X'3801'	Start of abort
X'4C01'	End of abort
X'5607'	Start unit of recovery
X'5613'	Recoverable in-doubt structure created
X'5614'	Recoverable in-doubt structure deleted

Sync Points with a Data-Propagation Manager

When using a data-propagation manager (such as the IMS DataPropagator™) to update DB2 UDB for z/OS databases synchronously with IMS DL/I databases, the updates to the DB2 UDB for z/OS databases are committed (or aborted) at the same time as the IMS updates. This provides consistency between the database management subsystems. IMS DB/DC, DCCTL, and DBCTL (BMP regions only) support the IMS Data Capture exit routine.

Restriction: In an IMS DBCTL environment, the data-propagation manager is only available for BMP regions.

For more information about the IMS DataPropagator, go to the following Web URL: http://www.ibm.com/software/data/dpropnr.

Chapter 5. Understanding IMS Logging

This chapter describes how and what IMS logs, and explains how logging affects IMS operations. See “Logging” on page 11 for an introduction to IMS logging.

The IMS Log Data Sets

This section describes the OLDS, WADS, SLDS, RLDS, RDS, RECON data set, the MVS log data set used for CQS, the CQS system checkpoint data set, and the CQS structure checkpoint data set.

Online Log Data Set

IMS uses the OLDS only in the online environment. The OLDS contains all the log records required for restart, recovery, and both batch and dynamic backout. The OLDS holds the log records until IMS archives them to the SLDS.

Define all of the OLDSs in the IMS procedure library (IMS.PROCLIB) using the `OLDSDEF` statement. The OLDS must be preallocated on a direct-access device. You can also dynamically allocate additional OLDSs while IMS is running by using the `/START OLDS` command.

IMS uses the Basic Sequential Access Method (BSAM) to write log records to the OLDS, and the Overflow Sequential Access Method (OSAM) to read the OLDS when IMS performs dynamic backout. Although referred to as a data set, the OLDS is actually made up of multiple data sets that wrap around, one to the other. You must allocate at least three, but no more than 100, data sets for the OLDS.

You can specify that the OLDS use *dual logging*, which is the duplication of information on two logs. When you use dual logging, an I/O error on either the primary or secondary data set causes IMS to close the nonerror OLDS and mark the error OLDS in the Recovery Control (RECON) data set as having an I/O error and a close error. IMS then continues logging with the next available pair. For dual logging, the minimum number of data sets is three pairs, and the maximum number is 100 pairs.

IMS uses as many OLDSs as you allocate. IMS issues a message each time it changes the current OLDS. This message identifies the OLDS being closed and the next OLDS to be used.

When any of the following events occur:

- IMS fills an OLDS
- An I/O error occurs
- You issue one of the following commands:
  - `/DBDUMP DB`
  - `/DBRECOVERY DB`
  - `UPDATE DB STOP(UPDATES) OPTION(FEOV)`
  - `UPDATE DB STOP(ACCESS) OPTION(FEOV)`

IMS does the following:

- Opens the next OLDS
- Notifies DBRC and the MTO that it is using a new OLDS

- Closes the current OLDS (both primary and secondary if you are using dual logging)

When IMS is using the last available OLDS, it alerts the MTO that no additional OLDS space is available. If archiving has not finished by the time all of the OLDSs are full, IMS waits until OLDS space becomes available. IMS will not log to an OLDS containing active data that is not yet archived. You must run the Log Archive utility to free the OLDS space. After IMS uses the last allocated OLDS, it reuses the first OLDS, if it has been archived.

You can use the /STOP command to stop and dynamically deallocate an OLDS. When stopped, that OLDS is no longer involved in the wraparound process.

Recommendation: Stop an OLDS when an error occurs that requires that OLDS to be recovered.

Restriction: You cannot stop the current OLDS. You cannot stop any OLDS when two or fewer OLDSs are currently available.

Similarly, you can use the /START command to start and dynamically allocate an OLDS. IMS retains the status of an OLDS (in-use, stopped, and so on) from one restart to the next.

Jobs to archive OLDSs might not complete in the order in which the OLDSs were created. For example, one OLDS might not yet be archived, but a subsequent OLDS might already be archived. When this occurs, IMS issues message DFS3259I and uses the next available OLDS.

The DBRC RECON data set contains information about the OLDSs for each IMS subsystem. Information in the RECON data set indicates whether an OLDS is available for use or contains active log data that must be archived.

Write-Ahead Data Set

IMS uses the WADS only in the online environment. The WADS contains a copy of committed log records that are in OLDS buffers, but that have not yet been written to the OLDS. In order to maximize log efficiency, IMS uses a log write-ahead function to write partially filled, padded blocks to the WADS (rather than the OLDS). IMS continually reuses WADS space after writing the appropriate log data to the OLDS.

The log write-ahead function ensures that all log records are on the log before IMS writes changes to a database. IMS updates a database in any of the following situations:

- When IMS needs to reuse the database buffer (if this is before commit)
- During commit
- During VSAM background write

If IMS fails, you use the log data in the WADS to complete the content of the OLDS and then close the OLDS as part of an IMS emergency restart or as an option of the Log Recovery utility. If you close the OLDS during emergency restart, you must include the WADS in use at the time of the failure.

You must preallocate and format the WADS on a DASD device that supports count key data (CKD) architecture. Format a WADS using the FORMAT WADS | ALL keywords on either the /NRESTART or /ERESTART commands. All WADSs must be on the same device type and should have the same space allocation. You can also dynamically allocate additional WADSs using the `/START WADS` command.

You can change any of the following specifications for the WADS during an IMS restart:

- Number of WADSs
- Sequence of WADSs
- WADS names
- Use of single or dual WADSs

Recommendation: To eliminate potential resource contention, place the WADS on a low-use device that is different from the device you use for the OLDS.

If you place the WADS on the same device as one of your OLDSs and use full-track blocking for the OLDS (in which a block is equal to a full track), the device should be able to handle infrequent OLDS seeks. Contention can still occur.

If the WADS and OLDS are on the same device, the Log Archive utility or dynamic backout can cause severe contention between an OLDS being archived and an active WADS.

System Log Data Set

IMS uses the SLDS in both the online and batch environments. In the online environment, an SLDS contains archived OLDS data. In the batch environment, an SLDS contains current log data.

Each execution of the Log Archive utility creates an SLDS. One SLDS can contain data from one or more OLDSs. You use an SLDS as input to the database recovery utilities (Database Recovery, Database Change Accumulation, and Batch Backout). You can also use an SLDS during an emergency restart of IMS. SLDSs can be stored on DASD, tape, or other mass storage.

DBRC maintains information about SLDSs in the RECON data set:

- For batch subsystems, DBRC maintains SLDS information in the PRILOG and SECLOG records.
- For online subsystems, DBRC maintains SLDS information in the PRILOG and SECLOG records only if you do not specify an RLDS when you run the Log Archive utility. Otherwise, DBRC maintains SLDS information in PRISLD and SECSLD records.

The Log Archive utility tells DBRC which OLDS it is archiving and which SLDS it is creating. The IMS online system can reuse OLDSs that have been archived.

Generally, you want to copy all the log records from the OLDS to the SLDS, but you can specify specific records. If you want to omit some types of log records from the SLDS in order to save space, include the `NOLOG` keyword when you run the Log Archive utility. The SLDS must always contain those records that might be needed for database recovery, batch backout, or IMS restart. The records that you can omit are:

X'10' Security violation records

X'45' Statistics records written during checkpoint

X'5F' Call trace record

X'67'	Communications (SNAP) trace records
X'69'	Unauthorized ID record (for 3275 display terminal)

IMS dynamically allocates an SLDS during IMS restart whenever log data required for restart read processing is not available from an OLDS. The OLDS might be unavailable because it has been archived, and because one of the following is true:

- The OLDS has been reused.
- The PRIOLDS and SECOLDS records have been deleted from the RECON data set.

To allow IMS to dynamically allocate SLDSs, you must specify the SLDS device type Dynamic Allocation macro (DFSMDA). DBRC provides the data set name and volume information required for dynamic allocation.

Recovery Log Data Set

When you run the Log Archive utility to create an SLDS, you can also request creation of an RLDS. The RLDS can be stored on DASD, tape, or other mass storage. The RLDS contains only the log records needed for database recovery:

X'24'	Database error records
X'3730'	Sync point records
X'4001'	Checkpoint records
X'4084'	
X'4098'	
X'42'	Checkpoint ID records
X'5612'	End of phase 2 commit records
X'5701'	Database begin update records
X'59'	Fast Path database change records
X'505x'	Database change records

IMS maintains RLDS information in the RECON data set in the PRILOG and SECLOG records. Whenever possible, DBRC uses the RLDS in place of SLDSs when creating JCL for the Database Recovery and Database Change Accumulation utilities. Using the RLDS rather than the SLDS is more efficient because the RLDS contains less information than the SLDS.

Restart Data Set

IMS writes system checkpoint information to the RDS. During each checkpoint, IMS creates or updates a checkpoint ID table; IMS uses this table during IMS restart to determine from which checkpoint to restart the system.

If, for any reason, the RDS is not available at restart, IMS can obtain the required checkpoint information from the log. However, using only the log could considerably lengthen the restart process.

Generally, you do not need to know the content of the RDS. IMS finds the information it needs in the RDS and uses it automatically during a restart.

RECON Data Set

DBRC automatically records information in the RECON data sets. Because both RECON data sets contain identical information, this book refers to them as a single data set.

IMS uses the RECON data set in many situations:

- During warm start and normal and emergency restarts. The RECON data set shows which data set—OLDS or SLDS—contains the most recent log data for each DBDS that you registered with DBRC.
- During logging, the RECON shows its latest status for the OLDS and whether the OLDS has been archived.
- For a recovery utility, DBRC selects the correct data sets.

MVS Log Data Set

The IMS Common Queue Server (CQS) records information about the data in the IMS shared queues in the MVS log data set. The MVS system logger serves the same purpose for CQS as the OLDS serves for IMS: it records all necessary information so CQS can recover structures in the coupling facility and restart after failure.

CQS writes log records for each pair of coupling facility list structures to a separate log stream. This log stream is shared among all the CQS subsystems that share the structure pair. MVS merges the log streams to make recovery possible.

Related Reading: For more information on the MVS system logger, see *MVS Programming: Sysplex Services Guide*.

CQS System Checkpoint Data Set

Each CQS subsystem maintains a system checkpoint data set for each structure pair in the coupling facility. Whenever the CQS subsystem takes a system checkpoint, it writes the checkpoint information to this data set.

The system checkpoint data sets are not shared among CQS subsystems.

Related Reading: For more information on this data set, see *IMS Version 9: Common Queue Server Guide and Reference*.

CQS Structure Recovery Data Set

Whenever a CQS subsystem takes a structure checkpoint, it writes checkpoint information to a structure recovery data set. CQS subsystems in a sysplex share the structure recovery data sets; there is one pair (two data sets) for each structure. CQS alternates between the two for each checkpoint.

Related Reading: For more information on this data set, see *IMS Version 9: Common Queue Server Guide and Reference*.

Archiving Log Records

For online systems, you can automatically or manually initiate archiving log records from the OLDS to the SLDS.

Automatic Archiving

If you have a large system with a lot of activity, you can minimize your intervention in the archiving process by using automatic archiving. It will eliminate the need to continually monitor logging to determine when to archive.

By default, IMS archives each OLDS when it is full. However, you can control how often archiving occurs by specifying how many OLDSs must be full before IMS archives them. Use the ARC= execution parameter or the AUTOARCH keyword of the /START command to control automatic archiving.

Recommendation: You must archive an OLDS before IMS can reuse it; be sure to archive frequently enough to avoid running out of OLDS space. If you run out of OLDS space, IMS waits until OLDS space becomes available.

Related Reading: For more information about the ARC= execution parameter, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*. For more information about the AUTOARCH keyword of the /START command, see *IMS Version 9: Command Reference*.

Manual Archiving

If you archive infrequently or at irregular intervals, you can initiate archiving yourself. Use the Log Archive utility (DFSUARC0) to archive logs manually. You can use the DBRC GENJCL.ARCHIVE command to produce JCL for the Log Archive utility. You can issue this command using the Recovery Control utility or an IMS online command (/RMGENJCL).

Archiving is also useful for batch systems to free disk space if your SLDSs are on disk. Use the Log Archive utility to copy an SLDS from DASD to tape. Because DASD and tape typically have different block sizes, the utility reblocks the log records while it copies them.

Copying an SLDS or an RLDS

You can use the Log Archive utility to copy an SLDS or RLDS to a new data set; however, you cannot use the GENJCL.ARCHIVE command to generate JCL to copy these data sets.

You can also use the Log Archive utility to create an RLDS or user data set. Use the Log Archive COPY control statement to do any of the following:

- Specify the user data sets to which you want log records copied.
- Determine which log records should be copied to a user data set.
- Specify that all log records required for database recovery should be copied to an RLDS.

Customizing Archiving

You can write user exit routines to process log records and copy certain log records to user data sets. For example, you can copy all records required for restarting Batch Message Processing programs (BMPs) to a user data set.

To customize archiving, specify the entry points for the exit routine using Log Archive utility control statements. IMS gives control to the exit routines when:

- The Log Archive utility is initialized.
- IMS reads the OLDS.

- The Log Archive utility terminates.

Related Reading: For more information about specifying entry points and running the Log Archive utility, see *IMS Version 9: Utilities Reference: System*. For more information about writing exit routines, refer to *IMS Version 9: Customization Guide*.

Tracing the Log

You can trace logging activity to diagnose performance problems or problems with IMS. As with all traces, the trade-off when using log tracing is between increased diagnostic capabilities and the overhead of running the trace. The overhead of tracing the log can be greater than running other types of trace, especially if you request an external trace, that is, if the trace itself is logged. But you can write the external trace to an external trace data set, and add no extra burden to the OLDS.

Specify log tracing in one of the following ways:

- Use the DLOG parameter on the OPTIONS control statement when you initialize IMS.
- Use the /TRACE command and omit the DLOG keyword. You can turn log tracing on and off and control whether it is to be logged to the OLDS or to an external trace data set.

Reducing Fast Path Logging

Because IMS holds updates for DEDBs in storage before writing entire VSAM control intervals (CIs), you can reduce the logging for Fast Path data. You reduce the log volume by logging only the changed data for each log record during replace (REPL) calls. You can reduce the logging only if the length of the segment remains unchanged.

Use the LGNR parameter of the IMS or DBC procedures to determine the maximum number of Fast Path DEDB buffer alterations that are to be held before IMS logs the entire VSAM control interval (CI). Use the Fast Path Log Analysis utility to evaluate the value you should use for the LGNR parameter.

Related Reading: For more information about the Fast Path Log Analysis utility, see *IMS Version 9: Utilities Reference: System*. For more information about the LGNR parameter, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

Using DBRC to Track Batch Job Logs

An IMS online subsystem always uses DBRC for tracking logs, but a batch job need not use DBRC. If you use DBRC for batch jobs, DBRC tracks which batch jobs create which SLDS.

Recommendation: Use DBRC for batch jobs to eliminate the need to manually keep track of batch SLDSs.

You do not need to create a log for read only (PROCOPT=G) batch jobs, but you do need to create a log for update jobs. For update jobs using DBRC, you cannot use DD NULLFILE or DD DUMMY in the JCL for the log data set.

Specify the use of DBRC during IMS system definition by using the DBRC keyword in the IMSCTRL macro. While IMS is running, you can use the DBRC= execution parameter in the DBBBATCH and DLIBATCH procedures to override the value specified during system definition. If you specify the FORCE keyword during system definition, you must use DBRC, except when you run the Log Archive (batch only), Log Recovery, or Batch Backout utilities.

Condensing an Accumulated SLDS or RLDS

You can use the IMS Database Change Accumulation utility (DFSUCUM0) to condense accumulated records in SLDSs or RLDSs.

As IMS runs, the number of SLDSs or RLDSs increases. You can use these data sets to recover a lost or damaged database, but to use them without change would be inefficient for the following reasons:

- Each SLDS or RLDS contains a record of activities of the entire IMS subsystem and of all the data sets for all the databases. Yet when you are recovering a database, you usually only recover a single data set. Thus, much of what is in the SLDS and RLDS does not apply.
- The SLDS and RLDS chronologically stores each change to any single database record. If a record changes 100 times since the last backup of the data set, the SLDS or RLDS includes all 100 changes. Yet, during recovery, you are only interested in the value the data had at the moment the data set was lost; the other 99 changes are irrelevant.

You can use the IMS Database Change Accumulation utility to sort through your accumulated SLDSs and RLDSs in advance and condense and streamline them. This utility:

- Picks out only those log records relating to recovery of databases
- Sorts these records by data set within a database
- Finds the most recent change in each part of an individual record

As the utility creates the change accumulation data set, IMS compresses repeated single characters, such as blanks and zeros. IMS expands the data again during recovery.

Running the Database Change Accumulation utility is not required, but using it periodically speeds database recovery. Alternatively, you can run the Database Change Accumulation utility only when the need for recovery arises (just before running the Database Recovery utility). Running these two utilities instead of just the Database Recovery utility can reduce the total time needed for recovery, depending on how much unaccumulated log information exists.

Related Reading: For more information on the Database Change Accumulation utility, see *IMS Version 9: Utilities Reference: Database and Transaction Manager*.

Input to the Database Change Accumulation Utility

In addition to using archived log data (SLDS and RLDS) as input to the Database Change Accumulation utility, you can also use a subset of the IMS log or a previous change accumulation data set. The utility writes the accumulated changes to a new change accumulation data set.

If the log data is on tape, you can specify all log volumes or a subset of log volumes as input to the Database Change Accumulation utility. When you specify a subset of log volumes, DBRC checks whether the subset is complete for each DBDS. A subset of log volumes is complete for a DBDS when all of the following conditions are true:

- The first volume in the subset is the volume with the first change to the DBDS since any of the following events occurred:
  - The last change accumulation.
  - The last image copy.
  - DBRC created the ALLOC record for this area (if the image copy was concurrent).
- The remaining volumes are in sequence.
- In a data-sharing environment, all logs containing changes for a DBDS are included.

Use the DBRC `GENJCL.CA` command to specify the subset of log volumes. You can request a specific number of log volumes either by volume (use the `VOLNUM` keyword) or by timestamp (use the `CATIME` keyword).

You can use a change accumulation data set as input to a later run of the Database Change Accumulation utility whether your subset of log volumes is complete or incomplete; however, you can use a change accumulation data set as input to the Database Recovery utility only if it represents a complete log subset.

Related Reading: For more information about the `GENJCL.CA` command, see *IMS Version 9: DBRC Guide and Reference*.

Changing Accumulation Groups

You can use DBRC to group DBDSs for which the Database Change Accumulation utility accumulates changes. These groups of DBDSs are called change accumulation groups.

You can define a change accumulation group by using the DBRC `INIT.CAGRP` command to write a CAGRP record in the RECON data set. Within this record, DBRC lists the DBDSs that make up the change accumulation group, identified by their database names and data set DD names. A change accumulation group can have up to a maximum of 2000 members.

Before you can use the `INIT.CAGRP` command to define the change accumulation group, you must identify each member to DBRC using an `INIT.DBDS` command. A DBDS can belong to only one change accumulation group.

You can add or delete members of a change accumulation group using the `CHANGE.CAGRP` command.

Restriction: Do not issue a `CHANGE.CAGRP` command while the Database Change Accumulation utility is running because you could damage your database integrity.

Defining Change Accumulation Data Sets for Future Use

You can define change accumulation data sets for future use for a given change accumulation group. Use the DBRC `INIT.CA` command to inform DBRC that these data sets exist, and specify the `REUSE` keyword on the `INIT.CAGRP` command when you define the group. The `GRPMAX` keyword of the `INIT.CAGRP` command determines how many change accumulation data sets you can define.

Reusing Change Accumulation Data Sets

To allow DBRC to reuse old change accumulation data sets, define a change accumulation group with the `REUSE` keyword and use the DBRC `GENJCL.CA` command to generate the JCL for the Database Change Accumulation utility job. The Database Change Accumulation utility reuses the oldest change accumulation data set when all available change accumulation data sets for a particular change accumulation group have been used and the maximum number of change accumulation data sets has been reached. Reusing a change accumulation data set means that DBRC uses its data set name, volumes, physical space, and record in the RECON data set as if they were for an empty change accumulation data set.

If you define a group with the `NOREUSE` keyword, rather than reuse the data set, DBRC deletes the RECON record for the oldest change accumulation data set. In this case, DBRC does not scratch the data set. You must scratch the data set or keep track of it, because DBRC is no longer aware of it.

Specifying Your Choices for the Log Data Sets

This section describes the choices you must make when defining the various data sets involved in logging:

- Online log data set
- Write-ahead data set
- System log data set
- Recovery log data set
- Restart data set

Related Reading: For information on defining CQS data sets, see *IMS Version 9: Common Queue Server Guide and Reference*.

Defining Online Log Data Sets

This section describes tasks related to defining OLDSs:

- Choosing dual or single OLDS logging
- Defining the number of OLDSs
- Defining space for each OLDS
- Defining a block size for the OLDS
- Defining devices for the OLDS
- Changing OLDS characteristics
- Defining OLDS buffers

Choosing Dual or Single OLDS Logging

IMS can log information to a single data set or to two identical data sets.

Definitions: *Single logging* uses a single data set, *dual logging* uses two data sets, where both data sets are identical. Whether you use single or dual logging, IMS writes information to sets of data sets, as described in “Online Log Data Set” on page 43.

Recommendation: Use dual logging whenever possible because the OLDS is of primary importance to system integrity. Specify dual logging in IMS.PROCLIB using the `OLDSDEF` statement.

With dual logging, IMS has two options when one of the OLDSs (of a pair) gets an I/O error. The first option is to discard the pair and switch to a good pair. This behavior is just like single logging mode.

The second option is to not have IMS discard the pair of OLDSs unless both pairs of OLDSs fail. Then, if all the good pairs of OLDSs fail, IMS can degrade to single logging mode and use the good OLDS from each pair. From this point, IMS behaves just like if it had been in single logging mode from the beginning. This option also simplifies operational procedures because, for example, you only need to run the Log Recovery utility to clean up the OLDSs when you use (single or dual) logging and a write error occurs.

Essentially, there are three logging states for IMS to run in:

- Single logging, which has no redundancy and has the highest maintenance when an error occurs. When only two good OLDSs remain, IMS will terminate.
- Dual logging with DEGRADE=NO, which behaves just like single logging except that data is written to two LOGS instead of one. And, like in single logging mode, if only two good pairs of OLDSs remain, IMS will terminate.
- Dual logging with DEGRADE=YES. In this case, when each pair of OLDSs has at least one write error, IMS will switch to single logging mode and start logging to whichever OLDSs in the remaining pairs are good.

In all three cases, when IMS gets to the point where it only has two data sets left to write to, it terminates. If, for some reason, IMS has no good data sets to write to, it will terminate with an ABENDU0616. For more information about log errors, see “Log Errors” on page 157.

Defining the Number of OLDSs

You must define at least three OLDSs (or OLDS pairs) to start IMS. However, you can define additional OLDSs (up to 100).

You must define the OLDSs to be used during initialization in the IMS.PROCLIB data set using the `OLDSDEF` statement. You can then dynamically allocate additional OLDSs. Specify the OLDSs you want to dynamically allocate using the DFSMDA macro; later you can use the `/START OLDS` command to add an OLDS.

Related Reading: For more information on the DFSMDA macro, see *IMS Version 9: Utilities Reference: System*.

When deciding how many OLDSs to define, consider the frequency of archiving and the amount of data you want to keep online. IMS reuses an OLDS only after IMS archives it. The number of OLDSs you define should be consistent with the frequency of archiving; if, for example, you archive frequently, you can probably plan on defining fewer OLDSs.

Recommendation: To avoid system failure, you should define more than the minimum number of OLDSs, even when using dual logging.

Defining Space for Each OLDS

When defining the size of each OLDS, you also need to consider the size and location (DASD or tape) of the SLDS. If the SLDS is on tape, consider the size of an SLDS volume and how often you intend to archive the OLDS. If the SLDS is on DASD, you should allocate enough space to contain all of the OLDSs to be archived. You might want to assign enough space to each OLDS so it fills an SLDS volume when it is archived. Or you might want to make each OLDS half the size of an SLDS volume, so you can initiate archive when two OLDSs are full.

Also, when defining the size of each OLDS, consider the amount of data you want to have online for doing such tasks as emergency restart or restarting a BMP. For example, if your system has a lot of activity and processes many transactions, consider defining larger OLDSs; then the records necessary for an emergency restart will more likely be online.

Defining a Block Size for the OLDS

You must choose the block size for the OLDS carefully because changing the size of the OLDS after it has been established requires that you stop online work, archive all OLDSs, and scratch and reallocate them to make sure their block sizes remain identical. When scratching and reallocating, you must delete OLDS entries from the DBRC RECON data set. After you change the block size of an OLDS, you can restart online work only from an SLDS.

Recommendation: Take a checkpoint soon after the restart so later restarts can use the new OLDS, rather than the SLDS.

The block size of each OLDS must be the same. The OLDS block size must meet all of the following requirements:

- The block size must be a multiple of 2048 bytes (2 KB).
- The block size must be at least 4 KB.

  Recommendation: Use a 6 KB minimum or use the length of the largest message segment.
- The block size must not exceed a maximum of 30720 bytes. This is the largest multiple of 2048 supported by BSAM.

The main factor that determines OLDS block size is the track size of the OLDS devices. The OLDS block size cannot exceed the device track size. You should pick the block size that maximizes the amount of log data per track (for example, full-track for 3330 or 3350 DASD; half-track for 3380 or 3390 DASD). Because IMS only writes full OLDS buffers to the OLDS, a large OLDS block size results in more efficient use of DASD space.

During initialization, IMS ensures that the block size specified for the OLDS is large enough to handle the maximum length log record. If the block size specified is too small, IMS discards the OLDS data set looks at the next OLDS. If, at the end of initialization, there are not at least three pairs of usable OLDSs, IMS terminates with an 0073 abend.

Table 3 gives some recommended OLDS block sizes for several DASD devices.

Table 3. OLDS Block Sizes for DASD Devices

Device Type	OLDS Block Size	Blocks per Track	Bytes of Log Data per Track
3330	12288	1	12288
3350	18432	1	18432
3380	22528	2	45056
3390	22528	2	47476

Table 4 on page 55 shows the relationship between the OLDS block size and the number of tracks (or cylinders) for 3380 and 3390 DASD.

Table 4. Relationship of OLDS Block Size and Number of Tracks

OLDS Block Size	OLDS on 3380	OLDS on 3390
6 KB	28 tracks or 2 cylinders	32 tracks or 3 cylinders
8 KB	25 tracks or 2 cylinders	30 tracks or 2 cylinders
10 KB	24 tracks or 2 cylinders	30 tracks or 2 cylinders
12 KB	21 tracks or 2 cylinders	28 tracks or 2 cylinders
14 KB	24 tracks or 2 cylinders	24 tracks or 2 cylinders
16 KB	18 tracks or 2 cylinders	27 tracks or 2 cylinders
18 KB	20 tracks or 2 cylinders	30 tracks or 2 cylinders
20 KB	22 tracks or 2 cylinders	22 tracks or 2 cylinders
22 KB	22 tracks or 2 cylinders	22 tracks or 2 cylinders
24 KB	13 tracks or 1 cylinder	26 tracks or 2 cylinders
26 KB	14 tracks or 1 cylinder	28 tracks or 2 cylinders
28 KB	15 tracks or 1 cylinder	15 tracks or 1 cylinder
30 KB	16 tracks or 2 cylinders	16 tracks or 2 cylinders

Defining Devices for the OLDS

You configure the OLDS so that a system failure that renders an OLDS inaccessible does not stop the entire system. If you define dual OLDS logging, define each data set in an OLDS pair on different devices and, if possible, on different control units and channels.

Example: Figure 4 on page 56 shows an OLDS configuration with multiple devices. In this configuration, if one device fails, you can still access the OLDSs on the other two devices. For example, if device B fails while logging to primary OLDS 1 and secondary OLDS 1, you can still use primary OLDS 1 as input to the Log Archive utility, and IMS continues logging using the next available pair of OLDSs: Primary OLDS 3 and Secondary OLDS 3. If both Primary OLDS 3 and Secondary OLDS 3 have not been archived, IMS continues logging with the next available pair, Primary OLDS 6 and Secondary OLDS 6. The primary OLDS steps from one device to the next and the sequence wraps around.

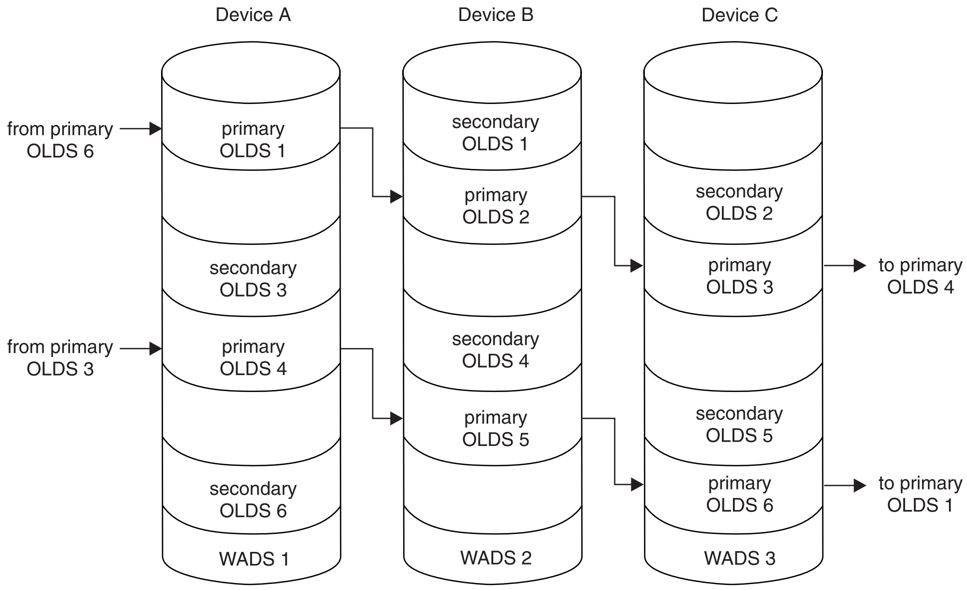


Figure 4. Sample OLDS Configuration

Changing OLDS Characteristics

Before you scratch and reallocate an OLDS that has been archived, you must delete the log control record in the DBRC RECON data set for this OLDS using the DBRC `DELETE.LOG OLDS(`*`dfsolpnn`*`) SSID (`*`mmmmmmmm`*`)` command.

You must delete the log control record for an OLDS that has been scratched and reallocated because the log control record only indicates that the OLDS has been archived. If you need this OLDS for IMS restart or batch backout, DBRC indicates to IMS to use this OLDS instead of the SLDS created by the archive job.

Related Reading: For information on using DBRC, see *IMS Version 9: DBRC Guide and Reference*.

Defining OLDS Buffers

You can define from 2 to 255 OLDS buffers (the default is 5). The number of buffers can be changed during IMS restart. If you change the number of buffers, each additional buffer might require additional tracks in the WADS. To calculate how many WADS tracks are required for each additional buffer, use the following formula:

```
Additional WADS tracks = (OLDS block size/2048) + 1
```

You might want to increase the number of OLDS buffers in the following circumstances:

- If IMS frequently waits for OLDS buffers
- If you have a high frequency of dynamic backout

Specify the number of OLDS buffers in IMS.PROCLIB using the `OLDSDEF` statement.

To provide virtual storage constraint relief, IMS allocates online log, batch log, and IMS monitor buffers above the 16 MB line. However, this relief is only available if you use MVS/ESA™ Data Facility Product 3.2 or later, and if you specify `LSO=S` on the `EXEC` statement for the IMS control region.

Defining Write-Ahead Data Sets

This section describes tasks related to defining WADSs:

- Choosing duel or single WADS logging
- Defining the number of WADSs
- Defining space for each WADS

Choosing Dual or Single WADS Logging

You can choose either single or dual logging for the WADS, just as you can for the OLDS. Using dual logging provides an alternate source of input to the OLDS in case an error occurs on one WADS while IMS uses it to close the OLDS. If one WADS is unusable, IMS uses the duplicate WADS. You can use single or dual WADS with either single or dual OLDS.

To request dual WADS logging, define more than one WADS and use the `WADS` keyword on the `EXEC` statement in the DBC procedure.

Defining the Number of WADSs

You must define at least one WADS on direct-access storage. You might want to define several WADSs (up to 10) as spares. IMS automatically switches to a spare WADS if the current WADS becomes unusable after an error occurs. If a write error occurs, logging to the WADS continues if:

- Single WADS logging is in effect and at least one WADS remains
- Dual WADS logging is in effect and at least two WADSs remain

Recommendation: If you use dual WADS logging, define at least three WADSs so that IMS can use the spare in place of either WADS if an error occurs; if you use single WADS logging, define at least two WADSs.

As when defining OLDS, make sure you have alternate ways of accessing WADS data. You might, for example, define each WADS on a different device (see Figure 4 on page 56).

Specify the number of WADSs using the `WADSDEF` statement in the IMS.PROCLIB data set.

Defining Space for Each WADS

IMS uses tracks in the WADS data set in groups. The size of a group depends on the size of the OLDS block size. Use the following formula to calculate the size of a group:

```
Number of tracks in a WADS group = (OLDS block size / 2 KB) + 1
```

Recommendation: Make the WADS large enough to hold at least one WADS track group for each OLDS block that fits on an OLDS track.

You can calculate the recommended minimum WADS sizes using the number of OLDS blocks per track (see Table 4 on page 55) and the following formula:

```
Minimum WADS size (in tracks) = (number of tracks in WADS group) X
(number of OLDS blocks per track)
```

The maximum amount of space you can define for each WADS is enough to contain 255 OLDS buffers. You can calculate the maximum number of WADS tracks that IMS will use using the following formula:

```
Maximum number of tracks = ([OLDS block size / 2 KB] + 1) X
(number of OLDS buffers)
```

Allocate the WADS in the range of the recommended minimum size and the maximum size. Most installations find that four to five cylinders are appropriate.

Defining System Log Data Sets

This section describes tasks related to defining SLDSs:

- Choosing dual or single SLDS logging
- Defining a block size for the SLDS

Choosing Dual or Single SLDS Logging

You can choose either single or dual logging for the SLDS, just as you can for the OLDS and WADS. When archiving to tape, you can force the primary and secondary volumes to contain the same data by specifying the number of log blocks per volume using the force-end-of-volume (FEOV) keyword of the SLDS control statement. When IMS writes the specified number of blocks, IMS forces end of volume on both the primary and secondary data sets.

To use dual logging, supply both a primary and secondary DD statement for each SLDS.

Defining a Block Size for the SLDS

The block size of the SLDS can differ from the block size of the OLDSs being archived. However, the block size of primary and secondary SLDS must be the same when using the FEOV keyword.

Specifying the Recovery Log Data Set

To use dual logging for the RLDS, supply both a primary and secondary DD statement for each RLDS in the JCL for the Log Archive utility.

Specifying the Restart Data Set

Allocate one cylinder of space for the RDS. Although the checkpoint ID table uses one track, IMS also writes other recovery information to the RDS.

Contents of the Log

IMS records activity on the OLDS. CQS records activity on an MVS log stream. Each different activity is recorded as a separate log record.

Generally, you do not need to know the content of log records. IMS and CQS identify the correct records and use them automatically when they perform recovery.

Related Reading: If you need to examine log records to solve a complex recovery problem, see *IMS Version 9: Diagnosis Guide and Reference*.

Log Reduction

As part of log reduction, IMS compresses log data. IMS compresses repeated single characters, such as blanks and zeros, in the segment data portions of log records. This compression applies to updates resulting from DL/I insert (ISRT), delete (DLET), and replace (REPL) calls.

The counterpart to compression is expansion. During backout and database recovery, IMS expands the data in the database buffer. For change accumulation data sets, IMS expands the data in the change accumulation record.

Using the Data Capture Exit Routine

IMS does not write log records for the Data Capture exit routine to show that it has been called, nor does IMS write the exit routine name in any log records. IMS does not differentiate between the application program and the exit routine. If you use the Data Capture exit routine extensively, your IMS system accounting and performance monitoring information is likely to show more system use for the application programs than they actually use.

Chapter 6. Making Database Backup Copies

This chapter explains how to make backup copies of your databases. It describes the utilities and how they affect IMS operations.

Related Reading: For information on making backup copies of the IMS message queues or system data sets, see “Backup” on page 14.

Image Copies (DFSUDMP0, DFSUICP0, and DFSUDMT0)

IMS provides three utilities you can use to make image copies:

- Database Image Copy utility (DFSUDMP0)
- Online Database Image Copy utility (DFSUICP0)
- Database Image Copy 2 utility (DFSUDMT0)

Restriction: You cannot run these utilities against a HALDB partition that is being reorganized by HALDB Online Reorganization (OLR).

These utilities create image copies of recoverable and nonrecoverable databases. The image copy utilities operate on data sets. With the first two utilities, if a database comprises multiple data sets or areas, you must supply the utility with multiple specifications. With the Database Image Copy 2 utility, however, you can copy multiple database data sets in one execution of the utility. You can specify a group name to represent the collection of database data sets that are to be copied in a single execution.

Use the `INIT.DB` and `CHANGE.DB` commands to identify recoverable databases to DBRC.

DBRC works similarly with all of the utilities. The output data sets from the utilities (except the Database Image Copy 2 utility) have the same format, and the rules for predefinition and reuse of image copy data sets apply to them all. The output data sets for the Database Image Copy 2 utility is in DFSMSdss™ dump format. These utilities call DBRC to verify the input (DBRC allows them to run only if the input is valid), and they call DBRC to record information in the RECON data set about the image copy data sets that they create. An image copy record in RECON data set has the same format regardless of which utility created its corresponding image copy data set. The Online Database Image Copy utility, however, has its own PDS member of skeletal JCL, and you use a separate command, `GENJCL.OIC`, to generate the job for the Online Database Image Copy utility.

Recommendation: Copy all data sets or areas for a database at the same time. When you recover a database to a prior state, you must recover all data sets belonging to the database, as well as all logically related databases (including those related by application processing) to the same point to avoid data integrity problems.

When using the image copy utilities, you have the option of creating one or more output image copies. Advantages of making two or more copies are:

- If an I/O error occurs on one copy, the utility continues to completion on the others.
- If one copy cannot be read, you can perform recovery using another.

The trade-off in deciding whether to make more than one copy is, of course, that performance of the image copy utilities is degraded by the time required to write the other copies.

The utilities enforce a minimum output record length of 64 bytes; thus, it is possible that an image copy of a database with a very short logical record length can require more space than the original database.

Recommendation: Take an image copy immediately after running batch jobs that do not use IMS logging. Even when pool sizes are identical, batch jobs are not repeatable if your system uses VSAM background write. Taking an image copy allows you to maintain the integrity of your database even though batch updates are not bit-for-bit repeatable. Do not rerun these batch jobs after an image copy because the database can become invalid when you apply change-accumulation or log tapes.

Concurrent Image Copies

IMS allows you to take an image copy of a database without taking that database offline. This concurrent image copy is often called a “fuzzy copy” because the copy represents the state of the database over a finite period of time rather than at an instant in time. IMS can update a database while making a concurrent image copy; some, all, or none of the updates might appear in the fuzzy image copy.

You can also use DFSMS and the Concurrent Copy feature of 3990 hardware to make copies of data sets while the data set is offline or online. This option will increase database availability.

The IMS Database Image Copy 2 utility (DFSUDMT0) calls DFSMS Concurrent Copy to make consistent image copies (that is, with no updates occurring while the utility is making the copy) or concurrent (fuzzy) image copies of an IMS database data set. You can make up to four copies at once.

Restrictions:

- You can only make a concurrent image copy of a database that is registered with DBRC.
- You can make copies of nonrecoverable databases, but they must be stopped before running the utility to make the image copy. You cannot take concurrent image copies of nonrecoverable databases while they are online because IMS does not log changes to them. The database itself becomes fuzzy if a fuzzy image copy of a nonrecoverable database is used to recover the database.
- Using the Database Image Copy utility (DFSUDMP0), you can only make concurrent image copies for OSAM and VSAM Entry Sequenced Data Set (ESDS) DBDSs; VSAM Key Sequenced Data Set (KSDS) DBDSs are not supported for concurrent image copy. If you use either of the other two image-copy utilities, you can create image copies of ESDSs or KSDSs.
- Databases and area data sets that are to be copied using the Database Image Copy 2 utility must reside on hardware that supports the DFSMS Concurrent Copy feature (such as a 3990 Storage Control Model 3, extended function with licensed internal code, or an equivalent device).
- If you do not use one of the IMS utilities to make image copies, you must keep track of the copies because IMS does not track user image copies.

Related Reading: For more information on the image copy utilities and how to specify a concurrent image copy, see *IMS Version 9: Utilities Reference: Database and Transaction Manager*.

Recovery after Image Copy

The Database Image Copy utility (DFSUDMP0) and the Database Image Copy 2 utility (DFSUDMT0) copy data sets for HISAM, HIDAM, HDAM, PHDAM, PHIDAM databases, and areas for DEDBs. When you perform a subsequent recovery, what you need depends on whether the copies are concurrent or not:

- For non-concurrent copies, you need only the image copy and those logs created after the database is restored to the online system.
- For concurrent copies, you need the image copy and logs created before and after the database is restored to the online system. DBRC helps you decide which logs you need.

Recommendation: When you run the Database Image Copy utility (DFSUDMP0) for databases and areas (without specifying the `CIC` on the `EXEC` statement for the utility), no other subsystem should update the databases. You can prevent updates to DL/I databases by issuing the `/DBDUMP DB` or `UPDATE DB STOP(ACCESS)` command. You can prevent updates to DEDB areas by issuing either the `/STOP AREA` or `UPDATE AREA STOP(SCHD)` command.

The Online Database Image Copy utility (DFSUICP0) runs as a BMP program. You can use it for HISAM, HIDAM, and HDAM databases only. If IMS updates these databases while the utility is running, IMS requires all logs for any subsequent recovery, including the log in use when you started the utility. IMS requires the logs because the image copy is not an image of the database at any one time.

HSSP Image Copies

If you use the image copy option of HSSP, IMS creates image copies of DEDB areas for you. The image copy contains the “after images” of the HSSP PCB, and is a fuzzy copy. IMS logs all other PCB database changes as usual. During database recovery, you must apply any concurrent updates since the start of the image copy process. A fuzzy image copy can be created if a non-HSSP region updated the same DEDB area during the image copy process.

Definition: *Fuzzy* means that there might have been concurrent updates during the image copy process.

If, however, the image copy process does not complete successfully, this data set is simply returned to the set of available HSSP image copy data sets.

IMS uses the QSAM access method of MVS to create HSSP image copies. The primary allocation of image copy data sets must be larger than (or equal to) the DEDB area data set size.

IMS treats HSSP image copies like concurrent image copies, so you can use the Database Recovery utility (DFSURDB0) without telling it that the image copy is an HSSP image copy.

Restriction: You can only make an HSSP image copy if a database is registered with DBRC. Furthermore, you must initialize the image copy data sets using the `INIT.IC` command.

Creating Image Copy Data Sets for Future Use

You can allocate image copy data sets before you need them. IMS automatically selects these data sets for use as output data sets when you use the GENJCL.IC or GENJCL.OIC command to generate a job for the Database Image Copy utility or Online Database Image Copy utility.

When you use the INIT.DBDS command to identify a DBDS or area in the RECON data set, you can specify any of the following keywords for that DBDS or area:

GENMAX
Use this keyword to specify how many image copy data sets you want DBRC to maintain information about for the DBDS or area.

You can maintain a certain number of image copy data sets for physical recovery. DBRC keeps a record of a specified number of the most recent image copy data sets. This number is the value you specify for the GENMAX keyword for this DBDS; duplicate image copy data sets are not included in this number.

RECOVPD
Use this keyword to maintain data for a certain period. For more information about this keyword, see “Recovery Period of Image Copy Data Sets.”

REUSE
Use this keyword to inform DBRC that you want to define image copy data sets and record them in the RECON data set for future use. DBRC records these available image copy data sets, with their names, unit types, and volume serial information, in the RECON data set, selects them during the processing of a GENJCL.IC command, and uses this information in the appropriate DD name of the job for the Database Image Copy utility.

NOREUSE
Use this keyword to inform DBRC that you do not want to use an existing data set, and that you want to provide the data set name for the output image copy data set that the Database Image Copy utility is to use. Specify the data set name in either the JCL partitioned data set (PDS) member that the DBRC uses to process the GENJCL.IC command or in the job you produce. When you specify NOREUSE, DBRC dynamically sets the unit type of the output image copy data set to the default unit type for the device (as specified in the INIT.RECON and CHANGE.RECON commands).

When the Database Image Copy utility uses an available image copy data set, DBRC adds a current time stamp to its record in the RECON data set.

Recovery Period of Image Copy Data Sets

The *recovery period* is the amount of time before the current date for which DBRC maintains recovery information in the RECON data set. For example, if the recovery period of a DBDS or area is 14 days, DBRC maintains sufficient recovery-generation information for at least 14 days.

You specify the recovery period using the GENMAX and RECOVPD keywords of the INIT.DBDS and CHANGE.DBDS commands. The following examples describe the affects of various DBRC keywords on the recovery period.

Example 1: Recovery Period of Image Copy Data Sets

The following table shows the DBRC keywords and their values for this example:

REUSE	GENMAX	RECOVPD
Y	Reached	Not Exceeded

If there are available image copies:

- IMS uses an available image copy.
- IMS issues message DSP0065I, indicating that the predefined image copy has been used.

If the number of predefined data sets equals the maximum number of generations:

- IMS cannot reuse the oldest image copy.
- IMS issues message DSP0063I, indicating that the image copy within the recovery period cannot be reused.
- IMS stops processing.

Example 2: Recovery Period of Image Copy Data Sets

The following table shows the DBRC keywords and their values for this example:

REUSE	GENMAX	RECOVPD
Y	Not Reached	Exceeded

Processing continues as described in “Reusing Image Copy Data Sets” on page 66, except that IMS uses the available image copy data set.

Example 3: Recovery Period of Image Copy Data Sets

The following table shows the DBRC keywords and their values for this example:

REUSE	GENMAX	RECOVPD
N	Reached	Exceeded

IMS deletes the oldest image copy data set record that exceeds the RECOVPD value.

Example 4: Recovery Period of Image Copy Data Sets

The following table shows the DBRC keywords and their values for this example:

REUSE	GENMAX	RECOVPD
N	Reached	Not Exceeded

If the number of image copy data sets used has reached the GENMAX value, IMS cannot delete the oldest image copy data sets within the recovery period. In this case:

- IMS issues message DSP0064I, indicating that an image copy data set within the recovery period cannot be deleted.
- Processing continues and DBRC records a new image copy data set in the RECON data set.

Example 5: Recovery Period of Image Copy Data Sets

The following table shows the DBRC keywords and their values for this example:

REUSE	GENMAX	RECOVPD
N	Not Reached	Exceeded

DBRC records a new image copy data set in the RECON data set. Even though the oldest image copy is beyond the recovery period, it will not be deleted because the GENMAX has not been reached.

Other Recovery Period Considerations

If you issue a CHANGE.DBDS command and specify a new value for GENMAX that is smaller than the existing value, IMS will record the value, regardless of whether the oldest image copies cannot be deleted because they are within the recovery period (RECOVPD).

When you issue the DELETE.IC command, IMS deletes any specified image copy data sets, regardless of whether the RECOVPD value has been exceeded.

Reusing Image Copy Data Sets

DBRC allows you to reuse old image copy data sets. The REUSE keyword of the INIT.DBDS command, in addition to allowing you to define image copy data sets for future use, allows DBRC to reuse image copy data sets. To reuse the image copy data set means that DBRC uses the same name, volume, physical space, and record in the RECON data set for the new image copy data set as for the old one.

When you run one of the image copy utilities, IMS automatically reuses the oldest image copy data set for a DBDS or area, if it can, when both of the following conditions are met:

- The RECON data set has records for a number of image copy data sets equal to the current GENMAX value. To see the current GENMAX value, use the LIST.DBDS command.
- The oldest image copy is beyond the recovery period.

When you use the GENJCL.IC command to generate the job for the Database Image Copy utility or Database Image Copy 2 utility, IMS automatically selects the image copy data set to be reused. If the number of image copy data sets is less than the GENMAX value, and all image copy data sets have been used, you must define more image copy data sets for the DBDS or area before running the Database Image Copy utility or Database Image Copy 2 utility. The number of image copy data sets should be greater than the GENMAX value if you want to use a recovery period.

If you do not allow IMS to reuse image copy data sets, but the GENMAX value has been reached and the RECOVPD has been exceeded, when you run the Database Image Copy utility or Database Image Copy 2 utility, DBRC selects a new image copy data set and deletes the record in the RECON data set with the oldest time stamp. IMS does not scratch the image copy data set itself. You must scratch the data set yourself or keep track of it, because DBRC is no longer aware of its existence.

HISAM Copies (DFSURUL0 and DFSURRL0)

Using the HISAM Reorganization Unload utility (DFSURUL0) to make backup copies of a database lets you process an entire HISAM database in one pass (the image copy utilities deal with single data sets or areas). The unload utility (DFSURUL0) also reorganizes the database as it copies it.

Because the unload utility (DFSURUL0) reorganizes the database, you must, before resuming normal online operations, reload the data set using the HISAM

Reorganization Reload utility (DFSURRL0), as shown in Figure 5. If you do not reload the data set, the logging done after unload but before reload reflects the old organization of the data set. Therefore, if you need to use that log to recover the data set in the future, the organizations will not match, and the data set's integrity will be destroyed.

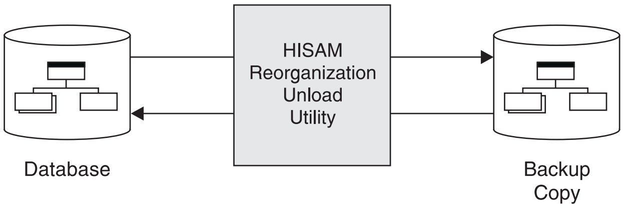


Figure 5. Making a Backup Copy with HISAM Unload

When using the HISAM utility to make a backup copy, you must reload immediately, or the actual database will not match the backup database. The reload utility (DFSURRL0) uses the output of its reorganization step as input to the reload step as if it were an image copy data set. After reorganizing a database, you must make an image copy of it before you can authorize the DBDS.

Before reorganizing a shared database, you must prevent other subsystems from being authorized during the reorganization process by:

- Issuing a global `/DBRECOVERY DB` or `UPDATE DB STOP(ACCESS)` command for the database to be reorganized. The `/DBRECOVERY DB GLOBAL` command prevents further authorizations except for reorganization and recovery utilities.
- Issuing the `CHANGE.DB` command with the `NOAUTH` keyword to manually update the RECON data set. This command prevents future authorizations except for the reorganization and recovery utilities. After the reorganization is complete, manually update the RECON data set by issuing the `CHANGE.DB` command with the `AUTH` keyword for the database that was just reorganized.

Recommendation: Ensure that recovery utilities do not run during the reorganization.

Non-standard Image Copy Data Sets

You can create the equivalent of an image copy data set using some means other than an IMS image copy utility. For example, you can use the Device Support Facility to make a copy of the volume on which a DBDS resides. DBRC does not automatically record the existence of these *nonstandard* image copy data sets in the RECON data set; you must use the `NOTIFY.UIC` command. If you do not record this information in the RECON data set, DBRC might misinterpret subsequent information about changes to the DBDS.

Restriction: When you use the `NOTIFY.UIC` command, you cannot specify the `REUSE` keyword of the `INIT.DBDS` command.

Before you recover a DBDS or DEDB area with a nonstandard image copy data set, you must restore the DBDS or DEDB area from the nonstandard image copy data set, and then issue the `NOTIFY.RECOV` command to notify DBRC that you restored the data set or area. If the time stamp of the nonstandard image copy is within the range of an existing time stamp recovery, the `NOTIFY.RECOV` command will fail. You then run the Database Recovery utility to add changes that occurred since the time stamp of the nonstandard image copy data set. DBRC provides and verifies JCL only for the sources of change records to be applied to the already-restored DBDS or DEDB area.

Because IMS does not use an image copy for this additional recovery, DBRC does not allow the Database Recovery utility to process any log that contains changes outside the recovery range. The recovery range is defined by the time-stamp recovery record's `RECOV TO` (image copy time) and `RUNTIME` values.

Recommendation: Close the database using the `/DBRECOVERY` command (without the `NOFEOV` keyword) or the `UPDATE DB STOP(ACCESS) OPTION(FEOV)` command before running the image copy utility.

Frequency and Retention for Backup Copies

You must consider two things when making backup copies of databases and areas:

- How frequently to make new copies
- How long to keep old (back-level) copies

There are no precise answers to these questions. Generally, the more frequently you copy, the less time recovery will take. The farther back in time your old copies go, the farther back in time you can recover; remember that program logic errors are sometimes not discovered for weeks. On the other hand, making each new copy requires work, and each old copy you save uses additional resources.

The only firm guidelines are these:

- If you create a new database, immediately make a backup copy of it.
- If a database is composed of several data sets, be sure to copy all data sets at the same time.
- If you reorganize a database, immediately make a new backup copy of it.

  Exception: It is not necessary to make backup copies after DEDB online reorganizations.

Chapter 7. Backout

Backward recovery or *backout* is one of the two major types of recovery. Using backout allows you to remove incorrect or unwanted changes from existing information or work.

The three types of backout are:

- Dynamic backout, which the system automatically performs
- Database batch backout, which you initiate using the Database Batch Backout utility (DFSBBO00)
- Backout during emergency restart, which the system automatically performs

Dynamic Backout

IMS automatically backs out changes to a database when any of the following events occur:

- An application program terminates abnormally.
- An application program issues a rollback call (ROLL or ROLB), or ROLS call without a token.
- An application program tries to access an unavailable database but has not issued the INIT call.
- A deadlock occurs.

In batch, you can specify (in the JCL) that IMS automatically dynamically back out changes if a batch application program abends, issues a ROLB call, or issues a ROLS call without a token. In this case, the log data set must be on DASD.

Dynamic Backouts and Commit Points

During dynamic backout, IMS backs out all database changes made by an application program since its last commit point. A commit point (or sync point) occurs in a batch or BMP program when the program issues a CHKP call. Commit points for message-driven application programs depend on the transaction mode (as specified by the MODE parameter of the TRANSACT macro).

Frequent commit points decrease performance. However, they also:

- Allow IMS to send output messages (replies) earlier
- Reduce the time required for emergency restart and database recovery
- Help avoid storage shortages for locking
- Help avoid reading SLDSs for backout, which decreases performance

Application programs can be batch oriented (non-message-driven BMPs) and choose when to commit their database changes using the CHKP call.

The following kinds of programs can issue a rollback (ROLB) call:

- Message-driven application programs that establish a commit point every time they attempt to get an input message
- Non-message-driven batch-oriented application programs

If an application program uses Fast Path resources or an external subsystem, IMS can issue an internal rollback call to back out the data to the last commit point. The application program receives status code FD as a result of this call.

IMS performs the following processing in response to a ROLB call:

- Backs out and releases locks for all database changes that the program has made since its most recent sync point
- Cancels all output messages since the program's most recent sync point
- Returns the first segment of the first input message since the most recent commit point to the application program, if an input message is still in process at this point, and if the call provides an I/O area

When IMS completes ROLB processing, the application program resumes processing from its last commit point.

When IMS initiates dynamic backout because of an application program abend, in many cases IMS stops both the transaction and application program. The MTO can restart the transaction or the application program. If you restart the application program before determining the exact cause of the abend, the program can abend again. However, if you restart the transaction, queuing of transactions continues.

Consider the transaction mode in deciding whether the MTO should restart transactions:

- If you restart response-mode transactions, and IMS subsequently enqueues new messages, any terminal entering the transaction is locked because no response is received.
- If you do not restart response-mode transactions, the terminal operator receives a message noting that IMS has stopped the transaction, but the originating terminal is not locked.
- If the transaction is not a response-mode transaction, you can restart it to allow terminal operators to continue entry. However, establish procedures in this case to warn terminal operators that they might not receive a response for some time.

When an application program abends, IMS issues message DFS554A to the master terminal. IMS does not issue this message for MPP regions when a resource shortage that is not expected to last long (such as short-term locks) occurs. In this case, IMS backs out the application program and places the input message back on the queue for rescheduling. When a BMP abends, IMS always issues message DFS554A because the MVS operator must restart BMPs.

Message DFS554A identifies:

- The application program (PSB) name
- The transaction name
- The system or user completion codes
- The input logical terminal name
- Whether the program or transaction is stopped

IMS also sends message DFS555I to the input terminal or master terminal when the application program abends while processing an input message. This error message means that IMS has discarded the last input message the application was processing.

The DFS555I message contains:

- The system or user completion codes
- Up to the first 78 characters of the input message
- The time and date

Related Reading See *IMS Version 9: Messages and Codes, Volume 2* for more information about these two messages.

Dynamic Backout in Batch

In a batch environment, when the SLDS is on DASD, you can request that IMS perform dynamic backout if IMS pseudo-abends or if the application program issues a ROLB call by specifying BKO=Y in the JCL. In this case, IMS performs backout to the last program sync point. Abend U0828 does not occur in batch if you specify BKO=Y.

Database Batch Backout

You can use the Batch Backout utility (DFSBBO00) to remove database changes made by IMS batch jobs and online programs. Because it is possible (in an online IMS subsystem) to have more than one dependent region using the same PSB, the utility might back out changes from several dependent regions when you execute batch backout against an online log.

You can use the Batch Backout utility to back out changes to the last checkpoint a batch job did not complete normally. If the batch region does not use data sharing, and if it is not a BMP, you can use the CHKPT control statement on the JCL for the Batch Backout utility to back out changes for the batch region to any valid checkpoint, whether or not it completed normally. Do not specify the CHKPT control statement when backing out changes for BMPs; the utility either rejects or ignores it.

The Batch Backout utility reads the log in the forward direction. Regardless of whether the backout is to the last checkpoint or to a specified checkpoint, the utility tries to back out all changes to all databases occurring between that checkpoint and the end of the log. When backing out BMPs, the utility always backs them out to the last checkpoint on the log.

In an IMS DBCTL environment, the Batch Backout utility backs out all in-flight updates, but only backs out in-doubt updates if you specify the COLDSTART option.

If dynamic backout fails or if backout during emergency restart fails, IMS stops the databases for which backout did not complete, and retries the backouts when you restart the databases.

When to Use the Batch Backout Utility

Use the Batch Backout utility in any of the following circumstances:

- If a batch job fails and you did not request automatic dynamic backout.
- If a batch job fails and you requested dynamic backout, but the failure was not a pseudo-abend.
- If you perform an emergency restart of IMS specifying the NOBMP option, and BMPs were active at the time of failure, you must run the utility once for each BMP. If you run the utility before restart completes, you must use the ACTIVE control statement.
- If you perform an emergency restart of IMS specifying the COLDBASE option, you must run batch backout once for each PSB (including MPPs, BMPs, and mixed mode IFPs) used by an application program that updates a full-function DL/I database, if the PSB was active at the time of the failure.

  You should also run batch backout for each PSB for which there is an in-doubt unit of recovery (UOR). Doing so puts the full-function databases in the same state as the Fast Path databases that are not updated until the commit point. If you choose not to back out an unresolved in-doubt UOR, you must remove it from the RECON backout record.

If you want to initiate these backouts before restart completes, you must specify the `COLDSTART` control statement for the Batch Backout utility. You do not need to specify the `COLDSTART` statement for backouts performed after restart.

DBRC protects all registered databases affected by in-flight and in-doubt UORs from access by other application programs until their backouts complete, even if the backout does not complete until after a cold start. You must ensure that programs do not access any unregistered databases before their backouts are complete.

Before issuing a `/ERESTART COLDSYS` command, you must run batch backout at least once using the `COLDSTART` or `ACTIVE` control statement. These control statements provide DBRC with the information necessary to protect registered databases in need of backout from erroneous access until the backouts are done.

If a failure occurs in one of the online backouts in a DB/DC or DBCTL subsystem that causes IMS to defer the backout:

1. Resolve the original problem.
2. Issue the `/START DB` or `UPDATE DB START(ACCESS)` command.
3. Complete the backout with a partial (restartable) backout.

In some cases, the restartable backout can also fail because the information it requires has not been saved. In those cases, you must use the Batch Backout utility. You must also use the utility if IMS defers one of the backout failures past a cold start.

System Failure during Backout

If a system failure occurs during backout, execute the Batch Backout utility again. Save the input logs and the output log from the successful backout run as input to the Database Change Accumulation or Database Recovery utility.

DL/I I/O Errors during Backout

IMS handles I/O errors for DL/I databases in the same way for the following kinds of backout:

- Dynamic backout
- Batch backout
- Emergency restart backout

Errors during Dynamic Backout

If a database I/O error occurs during backout, IMS issues message `DFS451I`, creates an EEQE to record the event, and calls DBRC to flag the appropriate database entry in the RECON data set as needing backout. You should run the Database Backout utility to back out the appropriate database entries and to inform DBRC to reset the appropriate backout-needed flag and counter. DBRC does not authorize flagged databases for use until you run the Database Backout utility.

For write errors, IMS copies the buffer contents of the blocks or control intervals in error to virtual buffers pointed to by the EEQE. IMS also tries to write each buffer that has an outstanding error when the database is closed. IMS issues message

DFS0614I for each error buffer that it successfully writes or reads (if the EEQE is for a read error), and issues message DFS0615I when all outstanding errors have been resolved.

Related Reading: For more information on recovery of such errors, see "DL/I I/O Errors and Recovery" on page 165 and "Database Failures" on page 162.

If a database read error occurs during dynamic backout, IMS issues message DFS983I, stops only the database in error, and continues dynamic backout of other databases within the PSB. IMS writes a X'4C01' log record for each database that it successfully backs out, and a X'4C80' log record for each database that it stops because of a read error. IMS then allows you to recover any stopped database and back it out offline.

If dynamic backout fails, you must run batch backout to reconstruct the database.

Recovering from Errors during Dynamic Backout

Different kinds of failure require different recovery procedures. Specifically:

- If a database is unavailable during dynamic backout or emergency restart, but subsequently becomes available (for example, you bring it back online or repair a physical device), use a /START DB or UPDATE DB START(ACCESS) command to reschedule dynamic backout to back out the (now available) database.
- If the database remains unavailable (because, for example, the database has an error that you must recover), you must run the Database Recovery utility before you allow further processing against the database in error, and then run the Batch Backout utility. Batch backout recognizes the X'4C01' log records and does not try to back out changes for databases that were backed out successfully.
- In the online environment, dynamic backout can fail if the log records are not available. If the only log records available are from SLDS, the performance of dynamic backout is poor. Slow dynamic backout can happen in the following cases:
  - The OLDS containing the log record has been archived and reused.

    Recommendation: Make sure that you have defined enough OLDS space, and that programs reach sync points (using GU or CHKP calls) frequently enough to ensure that records needed for backout are available from OLDSs.
  - IMS stops a database during dynamic backout if a nonrecoverable read error occurs on the logs when single or dual logging is in effect.

    In this case, you must run the Log Recovery utility to get a valid log. The output log must terminate normally; use it as input to the Database Change Accumulation utility or the Database Recovery utility. After you have corrected the input error, execute the Batch Backout utility again to correct the databases.

Errors during Batch Backout

If an I/O error occurs during batch backout, the Batch Backout utility completes backout of all databases except for those affected by read errors. IMS handles these I/O errors in the same way that it handles them when they occur during dynamic backout.

Before the Batch Backout utility makes any updates, it builds database buffers for any outstanding I/O errors. It applies updates to these buffers without trying to write them until the database is closed.

If any I/O errors remain unresolved for a database that has been backed out, you must eventually do a forward recovery, but you do not need to rerun the Batch Backout utility if it completed successfully (IMS issues message DFS395I).

Errors on Log during Batch Backout

If an I/O error occurs on the input log, you should execute the Log Recovery utility to correct the error. The output log must terminate normally; keep this log as input to the Database Change Accumulation or Database Recovery utility. After you correct the input error, execute the Batch Backout utility again.

If the I/O error occurs on the output log, terminate the output log correctly. Then execute the Batch Backout utility again. Keep both output logs as input to the Database Change Accumulation or Database Recovery utility.

Errors during Emergency Restart Backout

IMS handles both read errors and failures to open a database during emergency restart in the same way as it does during dynamic backout. For read errors, however, other restart backouts for the same database can take place, even though a database is stopped.

If a nonrecoverable OLDS read error occurs during backout (on both OLDSs if dual logging is in effect), run the Log Recovery utility and attempt to restart IMS again.

During restart processing, but before initiating backout, IMS determines if it must close the OLDS using the WADS. If a nonrecoverable read error occurs on the WADS, committed log records might be lost. In this case, you must close and archive the OLDS, and then reconstruct the affected databases by performing forward recovery and batch backout.

Chapter 8. Starting or Restarting IMS

Starting or restarting an IMS subsystem means initializing the IMS control region. To restore the operating environment, the MTO must ensure the following are also initialized:

- All IRLMs
- All CQSs
- Any dependent regions (including utilities)
- All CSL address spaces (OM, RM, and SCI), if operating as an IMSplex using CSL
- Any connections to VTAM
- All communication lines and terminals

The term MTO used in this and subsequent chapters refers to the IMS master terminal operator for the DB/DC and DCCTL environments, and to the operator for the DBCTL environment. All commands shown assume that the command-recognition character is the forward slash (/).

Starting the CSL Address Spaces

If you are running IMS as an IMSplex with CSL, you need to start the CSL manager address spaces (OM, RM, and SCI) before starting the IMS control region. You can start the CSL manager address spaces using the MVS START command from an MVS system console or by using JCL.

Related Reading: See *IMS Version 9: Administration Guide: System* for more information about IMSplex operations.

Starting the IMS Control Region

You usually start the IMS control region with an MVS START command from an MVS system console. If, depending on the size of your installation, you have an alternate system console for the IMS MTO, be sure the MVS system operator relays all IMS messages (that is, those beginning with BPE, CQS, DFS™, DSP, DXR, and ELX) to the IMS MTO. If the MVS system console is physically separate from the IMS MTO, you should have a local telephone or paging line to enable communication between the MVS operator and the IMS MTO. The MTO should also have easy access to the operators responsible for DASD and tape units.

When the IMS control region is started, the system parameters for that particular instance of the system are listed in message DFS1929I. The system parameters are also listed in the job log. This information will help you understand the configuration of a particular system. Understanding the configuration will be helpful in diagnosing problems.

If you are restarting a control region and the CPLOG value has been changed from the value set by system generation, the value displayed in message DFS1929I might not be valid. Issue the /DISPLAY CPLOG command after the restart process has completed to display the valid CPLOG value.

For online IMS systems, the control region automatically starts the DBRC address space. If you are running an IMS DB/DC or DBCTL environment, you can specify `LSO=S` as an EXEC parameter for the control region to start the DL/I address space. The following rules apply:

- IMS initialization waits until there is a separate address space available for DBRC.
- You can start DBRC manually (if, for example, a JCL error prevented DBRC from starting).
- When you restart IMS, you can change the specification of `LSO=`; IMS does not use any prior specifications.

After the control region has started, the MTO enters an IMS command to start the system. Select one of three ways of starting IMS:

1. Cold start
2. Warm start (also referred to as normal restart)
3. Emergency restart

Your operating procedures should include guidelines for the MTO in selecting the right type of startup.

You can alternatively use automatic restart instead of having the MTO enter an IMS command to restart the system. Specify `AUTO=Y` in the control region JCL to request automatic restart. IMS selects a suitable checkpoint for restart and notifies the MTO. You cannot use automatic restart if you:

- Want to perform a cold start
- Need to specify any options on the restart command (such as formatting system data sets, changing security options, or restarting only one portion of a DB/DC system)
- Need to override DBRC after a failure in which DBRC is unable to mark the subsystem record in the RECON data set as abnormally terminated, such as in the case of a power, CPC, MVS, or DBRC failure

Setting the MVS TOD Clock

The MVS time-of-day (TOD) clock setting is critical to the integrity of the IMS log. The TOD clock is also critical to the proper functioning of IMS restart, data sharing, XRF tracking and takeover, and RSR tracking and takeover.

Recommendation: Do either one of the following:

- Use an External Time Reference (ETR) device. Be sure to set it to use Universal Coordinated Time (also known as Greenwich Mean Time) with the appropriate offset for your local time zone. For information on using an ETR, see *OS/390 MVS Initialization and Tuning Guide* or *OS/390 MVS Initialization and Tuning Reference*.
- Set the MVS TOD clock during MVS IPL using Universal Coordinated Time, then set the local time. See “Setting or Changing Local Time” on page 77.

Attention: During MVS IPL, do not set the TOD clock to a time and date **earlier** than the immediate prior shutdown or failure. Setting the TOD clock back has severe IMS database integrity and recovery implications. If you do set the TOD clock back, you must reallocate the OLDSs using a different block size, reinitialize the DBRC RECON data sets, take image copies of all DBDSs, and cold start IMS.

Setting or Changing Local Time

To set or change the local time, edit the CLOCKxx member (the `TIMEZONE` keyword) of the SYS1.PARMLIB data set to set the current offset for local time. Then do one of the following:

- If you have an ETR device, use it to set and change the time zone offset.
- If you do not have an ETR device, do one of the following to set or change the local time:
  - Use the `SET CLOCK` command to set the time zone offset.

    Using the `SET CLOCK` command does not allow you to set a precise value for your system's time zone offset. Consequently, using this method may cause IMS to report a local time that is inconsistent with the local time reported by the operating system. For more information, see "Considerations for Setting Local Time."
  - Re-IPL your system with the updated CLOCKxx member. This sets your time zone offset to the precise value specified in the member.

You can reset local time whenever necessary, for example when changing from standard time to daylight savings (or summer) time.

Considerations for Setting Local Time

IBM strongly recommends that you set your system's time zone offset—the difference between local time and Universal Coordinated Time (UTC)—to an integer multiple of fifteen minutes. This is consistent with world time zone definitions and ensures that the local time reported by IMS matches the local time reported by the operating system.

IMS will function properly even if the time zone offset is not a multiple of fifteen minutes. However, due to the way IMS stores time values internally, not following this recommendation may result in IMS reporting a local time that is inconsistent with the local time reported by the operating system.

More About IMS Time Stamps and Setting the Time Zone Offset

IMS stores all date/time stamps internally as 12 bytes in the following packed-decimal format:

```
YYYYDDDF HHMMSSth mijuAQQ$
```

The date and time fields (`YYYYDDDF` and `HHMSSthmiju`) are in Universal Coordinated Time (UTC). The `A` field consists of 4 bits of flags used internally by IMS. The `QQ$` field contains the signed packed-decimal local time zone offset for when the time stamp was created. To get the local time, IMS adds the time zone offset—stored in quarter hours (fifteen-minute intervals)—to the UTC stamp. For example, the time stamp:

```
2000353F 06420588 4242032D
```

represents the UTC date and time of:

```
06:42:05.884242 on day 353 of 2000 (December 18th)
```

The `QQ$` field of this time stamp is 32D, which represents a local time zone offset of minus 32 quarter hours (or minus 8 hours west of Greenwich). Thus, the local time represented by this time stamp is:

```
22:42:05.884242 on day 352 of 2000 (December 17th)
```

This format allows IMS internal time stamps to encode both the UTC time (to provide an always-incrementing time that is not subject to daylight savings time changes), and local time (to determine the “wall-clock” time for when the time stamp was generated). If your time zone offset is set to a multiple of fifteen minutes, IMS can accurately use the offset to convert from the UTC stored in the 12-byte time stamp into the local time.

In this example, the local time will be consistent with the time reported by the operating system. However, if your time zone offset is not set to a multiple of fifteen minutes, the local time derived from this conversion will not be consistent with the time reported by the operating system.

In the following example, the time zone offset is set to +01:55 (one hour and fifty-five minutes east of Greenwich). A time stamp is generated at 15:00:00.000000 UTC time (16:55:00.000000 local time, using the current time zone offset setting). IMS will construct an internal time stamp of:

```
2001240F 15000000 0000008C
```

The encoded time zone offset is 8C, or +8 quarter hours (2 hours). Because IMS cannot store a time zone offset with a precision greater than fifteen minutes, it rounds to the nearest fifteen minute boundary.

When this time stamp is converted back into local time, IMS reports a time of 17:00:00.000000, not 16:55:00.000000. This does not cause problems for IMS, which uses the more accurate UTC form of the time stamp internally. However, you will notice this rounding effect when you compare IMS-generated local times to the current operating system time. For example, if you issue an IMS checkpoint command on the system described above, the DFS058I message will indicate two different times, as shown below:

```
16:55:00 DFS058I 17:00:00 CHECKPOINT COMMAND IN PROGRESS
```

The first time value (16:55) is returned by the operating system; this value is generated by using the MVS time zone offset of +01:55. The second time value (17:00) is returned by IMS; this value is generated from an internal IMS time stamp that uses a time zone offset of +08 quarter hours (2 hours).

This behavior may also be encountered by application programs that compare the local time stamp in the I/O PCB with the value returned from the MVS TIME macro. The I/O PCB local time represents the time when the message was received by IMS. The data in this time field is derived from an internal IMS 12-byte time stamp stored with the message.

In the following example, a message is received by IMS at 12:42 UTC on the system described above. The local time is 14:37, based on the MVS time zone offset of +01:55. However, the time presented to the application by IMS in the I/O PCB local time field is 14:42, due to the rounding of the time zone offset.

Now, suppose the message is scheduled at 12:43 UTC, one minute after it arrived. If the application program issues an MVS TIME macro request, it will receive back a local time of 14:38. If the application were to compare this time to the I/O PCB time, it would appear as if the message was being processed before it was enqueued (enqueue time = 14:42; process time = 14:38). This is only a problem for an application program if it compares the I/O PCB time to a time generated from an operating system time service.

Performing a Cold Start

Perform a cold start when starting the system for the first time, or after changing the nucleus in a system definition. IMS never uses information from a previous shutdown as input to a cold start.

In a nonshared-queues environment, a cold start assumes empty message queues, so IMS discards any messages that exist. You must close the last OLDS used by the online system before you attempt a cold start. You should also archive the last OLDS before a cold start, so the OLDS is available if a database recovery is required.

In a shared-queues environment, IMS does not discard messages on shared queues. If you want to discard messages on shared queues, you must scratch the shared-queue structures on the coupling facility, and scratch and reallocate the CQS structure recovery data sets.

Performing a Warm Start (or Normal Restart)

A warm start (or normal restart) is the most common way of initializing IMS. And it is the recommended way of restarting after a controlled shutdown.

Use the `/NRESTART` command to perform a warm start or a normal restart. You can use the `FORMAT` parameter to format various system data sets (something you might want to do if you have reallocated any of the data sets). If you are running an IMS DB/DC or DCCTL environment, and you request reformatting of any of the message queue data sets, you must specify `BLDQ` on the `/NRESTART` command if IMS was previously shut down command included either the `DUMPQ` or `PURGE` keywords.

If the MTO needs to dequeue transactions after a warm start, use the `/DEQUEUE SUSPEND` command.

If you want to reload MSDBs during restart, use the `MSDBLOAD` keyword on the `/NRESTART` command.

Performing an Emergency Restart

After IMS terminates without a controlled shutdown, you must perform an emergency restart. The one exception is if an emergency restart fails; in this case, you must perform a cold start (use the `/ERESTART COLDSYS` command).

For an emergency restart, IMS restarts from the point of failure. Before restarting IMS, the MTO must know whether system data sets must be reallocated and reformatted during restart. Table 5 shows symptoms (a message or abend number) that indicate whether a system data set needs reallocation and reformatting.

Table 5. Symptoms Indicating System Data Set Reallocation Is Needed. **Key:** *N/A=Not Applicable*

System Data Set	Write Error	Read Error	Data Set Full	Open Error
WADS	DFS414I	DFS739I	N/A	DFS3256I
Message queue	U0759	U0759	U0758	DFS986A
RDS	DFS3127I	U0970	N/A	U0970
MSDB checkpoint or MSDB dump	DFS2718I, DFS2722I	N/A	N/A	DFS2681I, DFS2713I, DFS2714I

The following situations cause the abends shown in Table 5 on page 79.

U0758 An IMS internally generated /CHECKPOINT DUMPQ command failed after messages DFS206I, DFS207I, or DFS208I because one of the message queue data sets is full.

U0759 An error occurred while attempting to read or write the message queue data sets.

U0970 The RDS or checkpoint table was not initialized, or restart was unable to open the input log.

If the WADS has not previously been formatted, you should format it by specifying the FORMAT WADS keywords on the /ERESTART command. If you want to reformat any of the message queue data sets, specify the BLDQ keyword. If no checkpoint specifying either DUMPQ or SNAPQ has been taken since the last cold start, specify CHECKPOINT 0 on the /ERESTART command.

During Extended Specified Task Abnormal Exit (ESTAE) routine processing, IMS issues message DFS0617I if the RDS buffers have been purged and the RDS has been successfully closed. However, if you do not receive this message, you do not necessarily have to reformat or reallocate the RDS.

Recommendation: You should reformat the RDS if you receive message DFS3127I or abend U0970, either of which indicate a read or write problem with the RDS. However, if the error has nothing to do with the RDS, you should not reformat the RDS.

If errors occur on both MSDB checkpoint data sets:

1. Issue the /ERESTART command.
2. Issue the /DBDUMP DB or UPDATE DB STOP(UPDATES) command for the MSDBs.
3. Shut down the system (in a controlled way).
4. Run the MSDB Dump Recovery utility.
5. Restart the system using an /NRESTART command, including the MSDBLOAD keyword so that IMS can reload the MSDBs.

SLDS Input to Warm Start and Emergency Restart

SLDS input might be required to perform a warm start or emergency restart. When the SLDS is required for restart, IMS dynamically allocates it. IMS also deallocates any data sets previously allocated with the dynamic allocation macro (DFSMDA) with IMSLOGR or IMLOGR2 as the DD name for the TYPE=SLDS statement. DBRC provides the data set name and volume information for the dynamically allocated SLDSs. However, DBRC does not provide the unit type information, so a DFSMDA member with the name of IMSLOGR must provide the unit type.

In some cases, you may want to force IMS to read from the SLDS instead of the OLDS during restart—for example, to prevent IMS from reading an OLDS that you know is invalid or has been corrupted. To force IMS to read from the SLDS, you need to delete the OLDS entry from the PRIOLDS and SECOLDS records in the RECON data set. To delete an OLDS entry from the RECON data set, use the DELETE.LOG OLDS(*ddname*) command, where *ddname* is the name of the primary OLDS.

Specifying Security Options at Startup

Depending on your system definition, the MTO might be able to specify whether IMS activates various security options during startup. Table 6 shows the restart command options that affect security. These options can be used with the `/NRESTART` command or the `/ERESTART COLDSYS` command.

Table 6. Security Options on the /NRESTART or /ERE COLDSYS Command

System				
Security		**JCL EXEC**	**/NRE or /ERE**	
Macro Name	**Macro Options**	**Parameters**	**Command Input**	**System Security Level after Restart**
SECLVL	=NOSIGN	SGN=N	USER	Signon active. Single signon per user for static users.[1]
			TRANAUTH	Signon and transaction authorization active.
			CMDAUTH	Signon, static, and ETO command authorization active.
			CMDAUTHE	Signon and ETO command authorization active.
			MULTSIGN	Signon active. Multiple signon per user (static users).[1]
	=SIGNON	SGN=Y	NOUSER	Signon not active.
	=TRANAUTH	TRN=Y	NOUSER	Signon and transaction authorization not active.
			NOTRANAUTH	Transaction authorization not active. Signon active.
	=NOTRAN	TRN=N	TRANAUTH	Signon and transaction authorization active.
	=FORCTRAN	TRN=F		Signon and transaction authorization enforced.
	=FORCSIGN	SGN=F		Signon enforced.
		SGN=Z	SNGLSIGN	Signon active. Single signon per user (static users).[1]
		SGN=G		Signon enforced. Multiple signons per user (static users).[1]
TYPE	=RACFCMD	RCF=C	NOUSER	Signon and command authorization for ETO not active.
			NOCMDAUTHE	Command authorization for ETO not active.
			NOCMDAUTH	Command authorization for ETO/static users not active.
		RCF=S	NOUSER	Signon for ETO and static users not active.
				Command authorization for ETO/static users not active.
			NOCMDAUTHE	Command authorization for ETO/static users not active.
			NOCMDAUTH	Command authorization for ETO active.
			CMDAUTHE	Command authorization for static users not active.
	=NORACFCM	RCF=N	CMDAUTHE	Signon and command authorization for ETO active.
			CMDAUTH	Signon and command authorization for ETO and static users active.
TERMINAL	=NO		TERMINAL	Terminal security active.
	=YES		NOTERMINAL	Terminal security not active.
	=FORCE			Terminal security enforced (NOTERMINAL invalid).
PASSWD	=NO		PSWD	Password security active.
	=YES		NOPSWD	Password security not active.
	=FORCE			Password security enforced (NOPASSWORD invalid).
TRANCMD	=NO		TRANCMDS	Transaction command security active.
	=YES		NOTRANCMDS	Transaction command security not active.
	=FORCE			Transaction command security enforced (NOTRANCMDS invalid).

Note:

[1] This also applies to dynamic users signing on to ETO terminals where the user structure name is different from the user ID.

During restart, IMS always loads the security tables for signon, terminal, password, and transaction command security from the active IMS.MATRIX data set.

Starting the IRLM

If you are running an IMS DB/DC or DBCTL environment, start the IRLM from the MVS system console using the MVS `START` command. You must start IRLM subsystems before starting the IMS batch or online subsystem. The IMS online subsystem identifies itself to the IRLM subsystem during IMS initialization or restart.

If you are using the IRLM in a data-sharing environment, see Chapter 14, “Data Sharing,” on page 209.

Starting the CQS

In an IMS shared-queues environment, IMS starts the CQS subsystem during IMS startup or restart if the CQS subsystem is not already running. After CQS is running, the IMS online subsystem identifies itself to the CQS subsystem during IMS initialization or restart. You can use the MVS START command to start the CQS subsystem if you want to ensure CQS is running before you start IMS.

If you register CQS with the MVS Automatic Restart Manager, you do not have to restart CQS after a CQS failure. And because CQS and IMS are separate subsystems, you do not have to restart CQS after an IMS failure.

A CQS that supports only a resource structure must be started manually because IMS does not start this CQS.

Starting Dependent Regions

IMS dependent regions are MVS regions that are separate from the IMS control region, and that are logically dependent upon the IMS control region.

The following types of programs execute in dependent regions:

- IMS batch message programs (BMPs) and message processing programs (MPPs)
- IMS Fast Path (IFP) message-driven application programs
- Fast Path DEDB online utility programs
- Coordinator controller (CCTL) application programs (in the IMS DBCTL environment)

You can start IMS dependent regions from the MVS system console or the IMS master terminal. From the MVS system console, use the MVS START command to start a reader procedure (usually IMSRDR) to read the JCL for the region. From the IMS master terminal, use the IMS /START REGION command to start a region. If you do not specify a procedure name on the START or /START REGION command, IMS uses the IMSMSG procedure.

You should not start IMS message regions and Fast Path regions with automatic priority group (APG) initiators because IMS cannot control the dispatching priority of an APG-initiated job or uniquely identify such jobs. You might need to start and stop MVS initiators for IMS message regions and Fast Path regions that have APG initiators.

IMS can dynamically allocate up to 255 regions. The total number of regions includes the number of regions allocated during IMS system definition plus the number of dynamically allocated regions or CCTL threads.

Message Processing Regions

One of the jobs in the IMS.JOBS data set contains the message region JCL. You can define more than one set of JCL if necessary. The default job name is IMSMSG. Parameters on the JCL EXEC statement define the transaction classes that are eligible for processing in the region. After you start a message region, transactions associated with the transaction classes supported by the region can be scheduled. You can change these classes using the /ASSIGN TRAN or UPDATE TRAN SET(CLASS(*new_class_number*)) command after you have started the region.

Batch Message Processing Regions

You can start BMP regions from the MVS console or the IMS master terminal. Generally, you use the MVS console and an MVS reader procedure to start a BMP, unless the JCL for the job (using procedure IMSBATCH) is in the IMS.JOBS data set. A JCL EXEC parameter indicates that the job is a BMP.

Fast Path Message-Driven Regions

You can start IMS Fast Path message-driven regions from the MVS console or the IMS master terminal. Generally, you use the MVS console and an MVS reader procedure to start a Fast Path message-driven region, unless the JCL for the job (using procedure IMSFP) is in the IMS.JOBS data set. A JCL EXEC parameter indicates that the job is a Fast Path message-driven application program.

Java Dependent Regions

You can start a Java message-driven dependent region. This region, which is called a JMP region, is similar to an MPP region. The JMP region is started in the same manner as an MPP region. The default job name is IMSJMP.

You can start a Java non-message-driven dependent region. This region, which is called a JBP region, is similar to a BMP region. The JBP region is started in the same manner as a BMP region. The default job name is IMSJBP.

DEDB Online Utility Regions

You can start all DEDB online utilities using procedure FPUTIL. Specify the utility in the TYPE operand in the control data set.

CCTL Regions

Because a CCTL region is not part of an IMS subsystem, see your CCTL documentation for information on how to start the CCTL.

Starting Data Communications

For remote terminal operators to enter transactions, you must initialize the IMS data communications network (for the IMS DB/DC and DCCTL environments). The procedures are different for the different access methods and protocols:

- Virtual Telecommunications Access Method (VTAM)
- Basic Telecommunications Access Method (BTAM)
- Advanced Program-to-Program Communications (APPC/MVS)
- Intersystem Communication (ISC)

Connecting to the VTAM Network

Before establishing a session with IMS for any terminal, verify the following:

- VTAM and the Network Control Program (NCP) are active.
- Controllers and logical units are active. If they are not automatically activated by VTAM, the VTAM network operator can use the VTAM VARY command to activate them.
- Controllers are powered on, appropriately configured, initialized, and activated for VTAM and NCP.

Recommendation: Start VTAM before starting the IMS control region.

Use the /START DC command to establish communications between IMS and VTAM. You also need to use this command in either of the following situations:

- If VTAM is stopped and restarted while the IMS control region is running
- If you terminate communications between IMS and VTAM using the /STOP DC command

The /START DC command tells VTAM to pass queued logon requests to IMS for all terminals defined to VTAM as belonging to IMS. The /START DC command activates the following processes:

Initiating IMS data communication processing

Opening the VTAM access method control block (ACB)

Enabling the IMS VTAM logon exit routine

Opening the primary and secondary VTAM master terminals if these terminals were not opened during IMS initialization (due to the specification of VACBOPN=DELAY in the DFSDCxxx IMS.PROCLIB member)

If VTAM is active when IMS starts, the IMS VTAM ACB is opened; otherwise, the ACB is opened by the /START DC command. Any logon requests received by VTAM before the IMS /START DC command, but after the IMS VTAM ACB has been opened, are enqueued by VTAM until the /START DC command completes. If the queuing of logon requests by VTAM causes operational problems, the specification of VACOPN=DELAY in the DFSDCxxx PROLIB member can be used to delay the opening of the VTAM ACB. When the master terminals are generated to use VTAM, the master terminal operator will be unable to enter either the /NRE or /ERE commands (including /ERE BACKUP) or the /START DC command. These commands can be issued using automatic restart (AUTO=Y) or automated operations.

Before you can establish a session between IMS and a logical unit or node, the logical unit or node must be active, connected, enabled, and available to VTAM and started in IMS. You can activate logical units or nodes when VTAM starts using VTAM start options, or a VTAM operator can activate them using the VTAM VARY command.

Normally after you start IMS, all VTAM terminals have a started status. If you stop a VTAM terminal with an IMS /STOP command, the terminal remains stopped after an IMS warm start or emergency restart. You must use the /START NODE command before issuing the /OPNDST (open destination) command to reactivate the terminal.

You must establish sessions between terminals and IMS before anyone can use them to transmit data. Sessions can be initiated in one of the following ways:

- VTAM can automatically log them on (based on the VTAM terminal definition).
- The end user can log on.
- You can issue an IMS /OPNDST command (if they are not automatically logged on).
- You can issue a VTAM VARY command.

Connecting to the BTAM Network

IMS automatically starts communications between IMS and BTAM when you start IMS. After you start IMS, the telecommunication lines have a stopped status, but the BTAM physical terminals normally have a started status. If you stop a physical terminal with a /STOP command, the terminal remains stopped after an IMS warm start or emergency restart. You can start the telecommunication lines, and physical terminals if necessary, using the /START or /RSTART commands. Normally, you use the /START command after an IMS cold start and the /RSTART command after a warm start. The /RSTART command starts the terminals so that they have the same mode they had when they were stopped at the previous system shutdown.

Connecting to APPC/MVS

Before an LU 6.2 conversation can be established with IMS, the following must be active: APPC/MVS, VTAM, and NCP.

Recommendation: Start APPC/MVS and VTAM before starting IMS.

When you specify APPC=Y in the IMS procedure, IMS automatically starts communication between IMS and APPC/MVS. If you specify APPC=N in the IMS procedure, you can use the /START APPC command to establish communications.

You also need to use the /START APPC command in either of the following situations:

- If APPC/MVS is stopped and restarted while IMS is running
- If you terminate communication between IMS and APPC/MVS using a /STOP APPC command

Connecting ISC Sessions from CICS to IMS

You can connect CICS ISC sessions to IMS in one of the following ways:

- You can initiate the session explicitly by specifying AUTOCONNECT YES on the DEFINE CONNECTION command in the CICS system definition (CSD) utility, or by specifying CONNECT=AUTO in the DFHTCT TYPE=TERMINAL program. When CICS is started, it attempts to establish all sessions for which AUTOCONNECT is specified.

  In order for session to be initiated, IMS must be active when CICS is started.
- The CICS operator can initiate a session by entering the following command:

  ```
  CEMT SET TERMINAL(termid) [ACQUIRED | COLDACQ | RELEASED]
  ```

  where *termid* is the four-character session name (defined on the DEFINE SESSIONS command in the CSD utility), or the TRMIDNT in the DFHTCT TYPE=TERMINAL program. The following describes the effects of the keywords for the CEMT SET TERMINAL command:

  ACQUIRED
  Specifies normal resynchronization with the IMS half-session. CICS is in session with the logical unit represented by the terminal.

  COLDACQ
  Specifies that no resynchronization is done.

  COLDACQ is a special form of ACQUIRED, in which no resynchronization is required. If the previous session abended, using COLDACQ overrides CICS integrity control, which can lead to integrity problems. Also, check the CSMT log for an activity keypoint (similar to an IMS checkpoint) after the restart of a session following a CICS failure. If there is no activity keypoint, reissue the CEMT SET TERMINAL(*termid*) COLDACQ command after the next emergency restart.

  RELEASED
  Specifies that CICS is not in session with the logical unit represented by the terminal.

Specifying RELEASED for a terminal that has an active session causes the session to be terminated. Running transactions are allowed to finish unless you also specify PURGE or FORCEPURGE.

The display area of the CEMT [INQUIRE | SET] TERMINAL command shows the status of the CICS half-session. If the session initiates successfully, the status changes from REL (released) to ACQ (acquired). If the session does not initiate successfully, CICS writes an error message to the transient data destination CSMT.

- An application program can initiate a session implicitly by using the ALLOCATE command. A program can use the ALLOCATE command only for the SEND/RECEIVE interface.

  Recommendation: Use the ALLOCATE SYSID form of the command because it allows CICS to select an available session.

  If a session is not immediately available, control is returned to the application program if either of the following occur:

  – The program issues a HANDLE CONDITION command.

  – The program specifies NOQUEUE on the ALLOCATE command.

  Otherwise, the command is queued until a session becomes available.

  Each of the following conditions causes a session to be “not immediately available”.

  – All sessions to the specified system (or the specified session) are in use.

  – The only available sessions are not bound.

  – The only available sessions are contention losers.

  Related Reading: For additional information on the ALLOCATE command and subsequent processing, see *CICS-Supplied Transactions*.
- CICS automatic task initiation (ATI) can initiate a session.

To determine the primary and secondary RU sizes, CICS uses the values for the RECEIVESIZE and SENDSIZE parameters on the DEFINE CONNECTION | SESSION command in the CSD utility or the values for the RUSIZE and BUFFER parameters on the DFHTCT TYPE=TERMINAL | SYSTEM program.

Restart after Shutting Down a Component of IMS

After performing the necessary recovery, you can reactivate the particular component you shut down or detached. For example, if you take a database offline because of an I/O error and then recover it, you can again make it available to applications.

Use the appropriate IMS command to restart the component. Most often, this command is some form of the /START or UPDATE command. For example, use /START DATABASE or UPDATE DB START(ACCESS) to restart a database or /START PROGRAM to restart an application program.

A list of restart commands and their corresponding shutdown commands is shown in Table 7 on page 87.

Table 7. Commands for Restarting Components of IMS

Restart Command	Corresponding Shutdown Command	Affected Components
/OPNDST	/CLSDST	VTAM session
/START	/STOP /IDLE /DBDUMP /DBRECOVERY	Areas, databases, VTAM logons, lines, logical terminals, VTAM nodes, OLDS, programs, regions, external subsystem connection, transactions, users
/RSTART	/IDLE	Lines, physical terminals, logical links, VTAM nodes
/RELEASE	/HOLD	Conversations
/UNLOCK	/LOCK	Databases, logical terminals, nodes, programs, physical terminals, transactions
UPDATE	UPDATE	Areas, data groups, databases, and transactions. See Table 66 on page 381 for the appropriate keywords.

Restart after Shutting Down IMS

You can restart IMS after you have done any necessary recovery. You can perform three kinds of restart:

- *Normal restart* initializes the IMS subsystem without reference to any previous execution of IMS (a cold start), or restarts a system that was terminated using a checkpoint command (a warm start).
- *Emergency restart* initializes an IMS subsystem after a system failure. During an emergency restart, IMS does the following:
  - Closes the OLDS from the WADS
  - Resets each active transaction to its last sync point
  - Resets each active region (BMPs, and CCTL or ODBA threads) to its last sync point

    You do not need to restart connected ODBA application programs or CCTLs. An IMS shutdown or failure simply disconnects from the ODBA application or CCTL and generally does not affect it.
  - Restores the databases to the last sync point
  - Restores local message queues to their condition at the time of failure

    An IMS failure does not generally affect shared message queues.

  You must manually restart regions and BMPs. However, when restarting a batch message processing region, IMS automatically determines the last BMP sync point if you specified `LAST` as an `EXEC` parameter. Otherwise, IMS tells you what the last BMP sync point was so you can specify it during restart.

If you have an IMS DBCTL standby environment, you can send the `/ERESTART` command to the pre-initialized DBCTL subsystem. Doing this is faster than starting the IMS job and waiting for the DBCTL-ready message before sending `/ERESTART` command.

- *Emergency restart with OVERRIDE* is necessary after a failure in which DBRC is unable to mark the SSYS record in the RECON data set as abnormally terminated, such as a power, CPC, MVS, or DBRC failure.

Automatic restart reduces MTO intervention and makes restart faster. With automatic restart, IMS automatically chooses the appropriate restart command (either `/NRESTART` or `/ERESTART`). If restart processing abnormally terminates **before** initial checkpoint, the appropriate automatic restart command is the same type (either `/NRE` or `/ERE`) as the aborted restart. The operator does not enter a restart command. Specify automatic restart by including `AUTO=Y` in the JCL.

You can implement normal restart procedures easily, and generally do not need to worry about it. However, an operator error could adversely affect system integrity in the following situations:

- Specifying cold start after a `/CHECKPOINT FREEZE` | `DUMPQ` | `PURGE` command when some messages are not completely processed
- Failing to supply the proper security options

Restriction: Do not change the IMS operating environment between shutdown and normal or emergency restart. For example, if you change from a nonshared-queues environment to a shared-queues environment, or change coupling facility structure names, IMS abends during normal or emergency restart.

If you change the operating environment, you should cold start the IMS subsystem; or you can use IMS online change (`/MODIFY` command) to make changes while IMS is running.

Cold Start

A cold start is required only when you first initialize IMS or after changing the nucleus during a system definition. You must also cold start IMS after scratching and reallocating shared-message-queue structures if these structures contained messages.

Request a cold start using the `/NRESTART CHECKPOINT 0` or `/ERESTART COLDSYS` command. When you specify the `COLDSYS` keyword, the `/ERESTART` command initiates a cold start, not an emergency restart.

In a nonshared-queues environment, a cold start assumes empty message queues: if any messages exist, IMS discards them. In a shared-queues environment, IMS does not discard messages on shared queues during a cold start. If any affinities exist for VTAM generic resource groups, IMS deletes them. IMS never uses information from a previous shutdown as input to a cold start.

Optionally, you can format the IMS message queue data sets, WADS, and RDS during a cold restart. During a cold start, IMS loads all control blocks from the system libraries.

Depending on what you specified during system definition, you need to specify whether IMS should activate the various security functions available in IMS. With a cold start, IMS loads the security tables for signon, terminal, password, and transaction command security from the active IMS.MATRIX data set.

In a shared-queues environment, any locked messages remain locked after the IMS cold start until IMS (or other Common Queue Server client) unlocks or deletes them.

If an emergency restart fails, you must perform a cold start using the `/ERE COLDSYS` command. This restart does not perform any database recovery; you must do that offline. You must close the last OLDS used online and also archive it because it might be required for these database recoveries.

Warm Start

A warm start is the most common way to reinitialize IMS, and is the recommended way of restarting after a controlled shutdown of IMS. Attempt a warm start **only** if IMS terminated in an orderly manner. Specify a warm start using the `/NRESTART` command.

Before starting IMS and entering the restart command, you must determine whether IMS should:

- Reload MSDBs
- Format specific system data sets during restart
- Activate specific security functions during restart

The following situations require you to reallocate and reformat specific system data sets after a controlled IMS shutdown:

- When a message queue data set is full (indicated by messages `DFS206`, `DFS207`, or `DFS208`) and IMS was shut down by an internally-generated `/CHECKPOINT DUMPQ` command.
- When a write error occurred on the RDS during checkpoint or restart (indicated by message `DFS3127I`).
- When you need to change the amount of space allocated to the WADS or a write error occurs on the WADS during normal operation. You should change the amount of space allocated to the WADS only after a controlled IMS shutdown. You can replace a WADS that encountered a write error during normal restart (`/NRE`) or emergency restart (`/ERE`). In any case, when IMS uses a new WADS, you must format it during restart.
- When read or write errors occur on one of the MSDB data sets during normal operation (indicated by messages `DFS2718I` or `DFS2722I`).

For other error situations that require reallocation and formatting of some or all of the system data sets, you must use the `/ERE` command.

Depending on what you specified during system definition, you need to specify whether IMS should activate the various security functions available in IMS. With a warm start, IMS loads the security tables for signon, terminal, password, and transaction command security from the active IMS.MATRIX data set.

During a warm start, IMS:

- Reestablishes the status of the control region using control blocks that were logged at termination
- Optionally restores local message queues if IMS logged them (you issued a `/CHE DUMPQ | PURGE` command)
- Restores in-doubt units of work so they can later be resolved

Unlike emergency restart, you might have to manually release any transactions on the suspend queue. You can do this by either of the following:

- Re-executing the transaction by issuing a `/START TRAN` *`tranname`* command. This command releases a specific transaction.
- Issuing a `/DEQUEUE SUSPEND` command. This command releases all transactions on the suspend queue.

When using XRF with VTAM Multinode Persistent Session (MNPS), IMS operates in an XRF environment and must use two ACBs: the Applid ACB and the MNPS ACB. During a controlled shutdown, IMS closes the MNPS ACB and the Applid ACB. Closing these ACBs terminates MNPS tracking for the MNPS ACB.

After a normal restart that follows this controlled shutdown, VTAM no longer maintains persistent sessions. Therefore, when IMS opens the MNPS ACB, it does not need to recover any previous sessions.

Related Reading: For more information about using the Extended Recovery Facility with MNPS, see *IMS Version 9: Administration Guide: System*.

Emergency Restart

IMS requires an emergency restart whenever it terminates without a controlled shutdown. IMS restarts from the point of failure, as described below. Initiate an emergency restart using the `/ERESTART` command.

With an emergency restart, IMS:

- Backs out DL/I in-flight units of work
- Applies committed but unwritten DEDB changes to the database
- Retains any in-doubt units of work for IMS subsystems connected to a CCTL or to Recoverable Resource Management Services (RRMS), the z/OS syncpoint manager.
- Resolves any unfinished External Subsystem Attach Facility (ESAF) units of work with external subsystems
- Releases locked messages in a shared-queues environment

Before starting IMS and entering the emergency restart command, you need to know the following:

- Whether system data sets must be reallocated and formatted during restart. See “Logging” on page 11 for information on how to allocate and format a system data set.
- Whether IMS has cleaned up its resources. You can check for message `DFS627I` or `DFS627W` to determine the previous ending status of the IMS resource termination manager. These messages tell you if IMS successfully completed resource cleanup processing. Successful resource cleanup guarantees IMS restart without requiring an MVS restart.

After an IMS failure, you can use the ESAF indoubt notification exit routine DFSFIDN0 to determine in-doubt units of work during restart of the failed IMS. You can resolve the work outside of IMS. For more information about this exit routine, see *IMS Version 9: Customization Guide*.

During an emergency restart, IMS restores the system to the status it had at the last system checkpoint. Databases and areas reflect their content at the last sync point for each dependent region. The local message queues reflect their content at the time of the failure. And the system has just taken a new checkpoint. When you restart the dependent regions, IMS reschedules the previously processing programs. IMS also releases any transactions on the suspend queue.

After an IMS failure when XRF is used with VTAM MNPS, VTAM continues to maintain persistent sessions until the timer specified by the `PSTIMER=` keyword (specified in DFSDCxxx IMS.PROCLIB member) expires. If there is no XRF takeover, then VTAM might continue to maintain active persistent sessions when the MNPS ACB is opened after emergency restart. If this occurs, IMS issues the command `SETLOGON OPTCD=NPERSIST`, which tells VTAM to drop all persistent sessions being maintained. This is consistent with emergency restart processing when the XRF USERVAR= keyword is used, which also has no session recovery. IMS then issues the command `SETLOGON OPTCD=PERSIST` to start the normal session persistence for all new sessions on the MNPS ACB.

If an emergency restart fails, you do not have to cold start the entire IMS subsystem. Table 8 shows the various restart commands and how they affect an IMS system.

Table 8. Effects of Emergency Restart Commands on an IMS Subsystem

Emergency Restart Command	Effect on the IMS DB or DBCTL Subsystem	Effect on the IMS TM or DCCTL Subsystem
/ERESTART COLDBASE	Cold restart	Emergency restart
/ERESTART COLDCOMM	Emergency restart	Cold restart
/ERESTART COLDSYS	Cold restart	Cold restart

- The `/ERE COLDBASE` command performs a cold start of the DB portion of an IMS DB/DC subsystem.

  If you use this command, you are responsible for the recovery of the databases. IMS does not redo committed DEDB updates, and does not back out in-flight updates for DL/I databases. IMS identifies and stops databases that have in-doubt data or that need backout or recovery. You can backout in-flight DL/I data by running the Database Batch Backout utility, for which you should close (and, optionally, archive) the OLDS.
- The `/ERE COLDCOMM` command performs a cold start of the TM portion of an IMS DB/DC subsystem.

  This command initializes the message queues, recovers DEDBs, reloads MSDBs, and backs out in-flight changes to DL/I databases. At the same time, IMS maintains all existing in-doubt data.
- The `/ERE COLDSYS` command allows you to cold start both DB and TM.

  This command is essentially a combination of both `/ERE COLDBASE` and `/ERE COLDCOMM`, but it does not read the OLDS. Therefore, closing the OLDS is essential. The processing described for both the`/ERE COLDBASE` and `/ERE COLDCOMM` commands also pertains to `/ERE COLDSYS`.

Of the three forms of emergency restart command, you will likely use`/ERE COLDCOMM` most often.

If an emergency restart (`/ERE`) fails and a subsequent emergency restart (`/ERE COLDBASE` or `/ERE COLDCOMM`) also fails, you must issue the `/ERE COLDSYS` command to restart IMS. You must also close (and optionally, archive) the last OLDS that was used online before performing this `/ERE COLDSYS` restart because the log must be available in case database recovery is necessary.

Restrictions: Do not add a Fast Path DBD to the active ACBLIB between an abnormal termination and an emergency restart. If you add a Fast Path DBD to the active ACBLIB after abnormal termination of IMS, it is not accessible after the emergency restart. If you make changes to a Fast Path DBD in the active ACBLIB after the last warm start or online change (/MODIFY), an emergency restart may fail with abend U0168.

If an IMS database encounters an I/O error and the Extended Specified Task Abnormal Exit (ESTAE) routine is not able to execute, then IMS does not create an EEQE and does not lock the error block. Consequently, an IMS emergency restart does not detect the bad block. In this situation, you might need to use the OVERRIDE keyword on the /ERE command or set the abnormal termination flag in the RECON data set for the failing subsystem.

Deferring Backout of BMP Programs to Speed Emergency Restart

After a system-wide failure, IMS backs out the interrupted online programs as it restarts. This backout must be complete before regular work can continue.

IMS lets you postpone backout of BMPs. This hastens restart if either of the following is true:

- A BMP had run a long time since its last commit point and needs lots of data backed out
- The program has a low priority and could have its restart postponed with no trouble

If you postpone BMP backout, you must run the Database Batch Backout utility some time before you rerun the BMP, or before you put the affected database back online (in case other programs need it).

If you register the affected database with DBRC, DBRC prevents programs from using that database before a backout is performed.

Use the NOBMP keyword on the /ERESTART command to request postponement of BMP backout.

Using MVS Automatic Restart Manager

You can use the MVS Automatic Restart Manager to restart a subsystem or a job after a failure of the subsystem or environment in which it is running. IMS supports the MVS Automatic Restart Manager in the DB/DC, DBCTL, DCCTL, and XRF environments. The IRLM and CQS subsystems also use the MVS Automatic Restart Manager. IMS batch environments and IMS utilities do not support the MVS Automatic Restart Manager.

After a failure, the MVS Automatic Restart Manager will restart only the IMS control region and the CQS subsystem (each independently of the other). Other regions (such as DLISAS and DBRC) are in turn restarted by the IMS control region, and IMS dependent regions are typically restarted using automation.

In an XRF environment, the active IMS subsystem is automatically de-registered from the MVS Automatic Restart Manager while the alternate IMS subsystem is in its tracking phase. Thus, the active subsystem is not be restarted after a failure; instead, the XRF alternate takes over as expected.

The MVS Automatic Restart Manager ignores AUTO=N in the IMS startup parameters.

If an IMS or CQS subsystem abends before it completes restart, it is de-registered from the MVS Automatic Restart Manager so that the abend can be fixed before attempting restart again.

If an IMS or CQS subsystem that is registered with the MVS Automatic Restart Manager is cancelled by the MVS operator (using the CANCEL command), that IMS subsystem is not restarted by the MVS Automatic Restart Manager unless the operator includes the ARMRESTART keyword on the command.

Related Reading: For more information on how to register an IMS subsystem with the MVS Automatic Restart Manager, see *IMS Version 9: Administration Guide: System*.

For information on how to register a CQS subsystem with the MVS Automatic Restart Manager, see *IMS Version 9: Common Queue Server Guide and Reference*.

Restarting BMPs

IMS does not automatically restart BMPs after a system failure. You need to start them yourself after IMS has completed its restart processing.

The way in which you restart a BMP depends on whether its last checkpoint records exist in the OLDS. When the last checkpoint records are in the OLDS, you can specify one of four values during restart:

- The 8-byte checkpoint ID
- The 30-byte restart parameter IIIIDDDHHMMSSTHMIJU+0000 from message DFS0540I, where:

  IIII
  Is the region ID

  DDD
  Is the day of the year

  HHMMSSTHMIJU+0000
  Is the time in hours, minutes, seconds, and tenths, hundredths, thousandths, ten-thousandths, hundred-thousandths, and millionths of a second, plus the local time-zone offset
- The value LAST
- Nulls (if you do not want to restart the BMP)

Specify this value as an EXEC parameter or place it in the I/O area. When you specify LAST, you must not change the BMP's job name, PSB name, or program name; IMS uses these names to locate checkpoint information for the BMP.

If the BMP's last checkpoint records are not in the OLDS, you must identify the desired starting point in the program's JCL (IMSLOGR DD statement) when you restart it. This starting point is the last intermediate checkpoint the program took before you shut the system down. When restarting a BMP from a checkpoint, provide the IMS log data set containing the information that was recorded at that checkpoint. The input data set can be an OLDS, an SLDS, or a user data set containing log records.

If the program issued XRST and CHKP calls, you can restart it from the last checkpoint by specifying the checkpoint ID in the JCL and supplying, as input, the SLDS created in the program's previous execution.

Restarting Batch Jobs

Before you restart a DL/I batch job, ensure all database changes have been backed out. IMS might have backed out the database changes dynamically if a pseudo-abend occurred and you specified `BKO=Y` in the JCL and the data set resides on DASD. Otherwise, run the Database Batch Backout utility.

To restart a batch job, specify the ID of its last checkpoint in the JCL and supply, as input, the SLDS created in the program's previous execution. The input is specified on the `IMSLOGR` DD statement.

x The batch job that is performing the restart must have the same job name as the
x original. If you do not use the same job name for the restart, IMS cannot find the
x checkpoint and the job fails with a U0102 error.

Reconnecting CCTLs or ODBA Application Programs

IMS DB system shutdown or failure generally does not cause the termination of a CCTL or ODBA application program, but it does disconnect the CCTL or ODBA application program. The CCTL or ODBA application program can automatically reconnect after an IMS system termination, or a CCTL operator can manually reconnect it.

Related Reading: For more information on reconnecting CCTLs, see *IMS Version 9: Customization Guide* or your CCTL documentation.

Chapter 9. Controlling IMS

Controlling IMS consists of:

- Monitoring the system
- Processing the IMS system log information
- Choosing tools for detailed monitoring
- Executing recovery-related functions
- Modifying and controlling system resources
- Gathering performance-related data
- Controlling log data set characteristics
- Formatting newly initialized volumes for OLDS
- Connecting and disconnecting subsystems

Monitoring the System

In order to gather problem determination and performance information, you need to monitor the status of the system on a regular schedule. For example, to determine if you should start an extra message region, you might monitor the status of the queues during peak load.

You can determine the current status of the system by issuing `/DISPLAY STATUS` or `QUERY DB STATUS` commands, specifying the appropriate keywords. You can monitor the status of the IRLM by using the MVS command `MODIFY` *irlmproc*`,STATUS`.

You can use the `/TRACE` or `UPDATE TRAN` command to help diagnose system operation problems. These commands turn on or off various IMS traces, which record use of IMS control blocks, message queue and I/O line buffers, and save area sets. IMS records trace information on the IMS log unless you request that the traces be recorded on an external trace data set. See "Using the Trace Facility" on page 102 for more information. You can also use `/TRACE` to trace locking activities and to start and stop the IMS monitor (see "Using the IMS Monitor" on page 98). For the appropriate keywords for the `/TRACE` or `UPDATE` command, see Table 66 on page 381.

Related Reading: For more information on monitoring, see *IMS Version 9: Administration Guide: Database Manager*.

Processing IMS System Log Information

The system log data sets are a basic source for statistics about the processing performed by the online system. Individual log record types contain data that can be analyzed in many ways. For example, you can select and format all activity pertaining to a specified user ID, or about IMS pools.

Using IMS System Log Utilities

IMS provides several utilities to assist with extracting log records from the system log. These utilities also assist with reducing and merging data that is spread across several log data sets. The following sections describe several of these utilities.

File Select and Formatting Print Program

If you want to examine message segments or database change activity in detail, you can use the IMS File Select and Formatting Print Program (DFSERA10). This utility prints the contents of log records contained in the OLDS, SLDS, or the CQS log stream. Each log record is presented as one or more segments of 32 bytes. The printed output gives each segment in both character and hexadecimal formats.

You can specify selection criteria for a subset of the records rather than printing all records. You can also specify a starting record number and the number of records to process. You can use an exit routine to customize the selection and formatting of the log records.

Although you can use the File Select and Formatting Print Program to copy entire input logs, you can more conveniently use the Log Archive utility (DFSUARC0). You use one or more SLDSs as input and specify a user data set as output. Also, you need to specify `DBRC=NO` in the `EXEC` statement to prevent DBRC from making entries in the RECON data set about your backup log. Making backup copies of the system log data sets can be useful to obtain an alternative input source for statistics and other monitoring activities occurring in parallel with production use of the system log.

Related Reading: For detailed information about this utility, see *IMS Version 9: Utilities Reference: System*.

Log Transaction Analysis Utility

In an IMS DB/DC or DCCTL environment, you can collect information about individual transactions, based on records on the system log, using the Log Transaction Analysis utility (DFSILTA0). Many events are tabulated in the Log Analysis report produced by this utility, including total response time, time on the input queue, processing time, and time on the output queue.

You can select a start time for the report tabulation; analysis begins at the first checkpoint after the start time. To control how much transaction activity is tabulated, you can specify an interval (in minutes) of elapsed time from the start time before the utility ends the tabulation, or you can relate the activity reported to a number of IMS checkpoints.

You can retitle a Log Analysis report or change the sequence in which the detailed transaction lines are printed. You can sort by transaction code or by any of the fields in the report. You can also suppress printing so that the output is stored on a DASD data set.

Using this utility, you can create an output data set, in system log format, that is a copy of all or part of the input system logs. By having a copy of the system log, you can monitor system activity without impacting the use of the OLDS for recovery.

Related Reading: For detailed information about this utility, see *IMS Version 9: Utilities Reference: System*.

Statistical Analysis Utility

In an IMS DB/DC or DCCTL environment, you can produce several summary reports using the IMS Statistical Analysis utility (DFSISTS0). You can use these reports to obtain actual transaction loads and response times for the system. The statistics produced are dependent on the input system log data sets. The following set of reports is produced:

- Telecommunication line and terminal (distributed traffic over 24-hour day)
- Transaction (distributed activity over 24-hour day)
- Transaction response

- Messages queued but not sent (listing by destination and by transaction code)
- Program-to-program messages (listing by destination and by transaction code)
- Application accounting
- IMS accounting

Related Reading: For detailed information about this utility, see *IMS Version 9: Utilities Reference: System*.

Log Summary Utility

In a DB/DC or DCCTL environment, you can use the Log Summary utility (DFSKSUM0) to create a summary of the records contained in the log.

Related Reading: For detailed information about this utility, see *IMS Version 9: Utilities Reference: System*.

Using the IMS Performance Analyzer for z/OS

The IMS Performance Analyzer (IMS PA), program number 5697-B89, gives you the information you need to increase your IMS system performance. IMS PA provides more than 30 reports that can help optimize IMS system and application performance.

Specifically, IMS PA:

- Offers an ISPF CUA® compliant user interface to build, maintain, and submit report requests
- Provides revised, enhanced, and new reports
- Supports IMS Versions 5, 6, and 7 from a single load library
- Allows the optional use of GDDM® for selected graphical reports
- Saves selected report data for reporting using PC tools
- Requires no dependency on GPAR

IMS PA produces a comprehensive set of reports (from management summaries to detailed program traces), organized by level of detail and area of analysis, to meet a broad range of IMS system analysis objectives, such as:

- **System performance evaluation.** IMS PA features help you monitor and evaluate IMS system performance on a daily basis. Management level summary reports express key values in terms of rates, ratios, and percentages. These reports help you use IMS PA for trend analysis, comparative analysis of systems, and evaluation of a system against installation standards. IMS PA's fast and efficient processing of monitor output data lets you produce long and frequent traces to obtain an accurate view of your IMS system.
- **System tuning.** Reports that help you enhance IMS system performance through system tuning are a key feature of IMS PA. Monitor output is summarized and categorized to help you rapidly identify problem areas. Detailed analysis reports help you investigate and evaluate these problem areas and also the effect of changes to the system.
- **Application and program evaluation.** IMS PA reports program activity in message processing or batch regions. IMS PA can be a valuable tool for evaluating existing applications and programs and validating whether new applications and programs conform to installation standards. Program activity reports and program traces add greatly to system documentation.

IMS PA produces alphanumerically collated report items in terms of ratios, rates, and percentages to facilitate comparison of results without additional computations.

Schedules in progress, including wait-for-input (WFI) and BMPs, are reported. Reports on IMS batch programs are also provided.

IMS PA is a functional replacement for IMS/VS Performance Analysis Reporting (IMSPARS) and IMS Monitor Summary and System Analysis Program (IMSASAP).

Related Reading: IBM also offers a number of other IMS database productivity tools. IMS Support Tools is a set of database performance enhancements for your IMS environment. These tools can help you automate and speed up your IMS utility operations. They can also assist you in analyzing, managing, recovering, and repairing your IMS databases. You can learn more about these tools on the Internet at http://www.ibm.com/software/data/ims/about/imstools/.

Choosing Tools for Detailed Monitoring

Many of the monitoring tools you can use to collect detailed data are also used for general diagnostics. The principal tool provided by IMS is the IMS Monitor, which allows you to monitor online subsystems. For a stand-alone IMS DB batch system driven by an SLDS, use the Database Batch Monitor. You can also use IMS PA, program isolation and lock traces, and the external trace facility.

For information about IMS PA, see “Using the IMS Performance Analyzer for z/OS” on page 97.

Using the IMS Monitor

The IMS Monitor collects data while the online IMS subsystem is running. It gathers information for all dispatch events and places it (in the form of Monitor records) on a sequential data set. You use the IMSMON DD statement in the IMS control region JCL to specify the IMS Monitor data set. IMS adds data to this data set when you activate the Monitor using the `/TRACE` command. The IMS MTO can start and stop the Monitor to obtain snapshots of the system at any time. But, remember that the IMS Monitor adds to system overhead and generates considerable amounts of data.

Controlling IMS Monitor Output

Plan to run the IMS Monitor for short intervals and to control its operation carefully. Shorter intervals also prevent the overall averaging of statistics, so that problems within the system can be more readily identified. The IMS Monitor’s output can be constrained by:

- Type of activity monitored
- Database or partition or area
- Dependent region
- Time interval

IMS Monitor Output Data Sets

The IMS Monitor output can be either a tape or a DASD data set. Using DASD eliminates the need to have a tape drive allocated to the online system. If you want to use the Monitor frequently, you might find that permanently allocated space for a DASD data set is convenient. One technique is to code `DISP=SHR` on the IMSMON DD statement so that the reports can be generated as each Monitor run is completed.

You must coordinate the report generation with the operator because each activation of the monitor writes over existing data. Although this does not occur for tape data sets, new volumes must be mounted. The volume is rewound, and a mount request is issued each time you start the Monitor.

Recommendations:

- Do not catalog Monitor data sets. The Monitor can produce multiple output volumes while IMS is running if the data sets are not cataloged.

  If you want to have IMS dynamically allocate the Monitor data set, do not include the IMSMON DD statement in the IMS control region JCL.
- Allow IMS to dynamically allocate IMS Monitor tape data sets. A tape drive is not permanently reserved for the control region for dynamically allocated data sets.

Related Reading: For details of how to specify dynamic allocation for the IMS Monitor data set, see *IMS Version 9: Utilities Reference: System*.

Selecting IMS Monitor Traces

After monitor requirements have been established, you can restrict the scope of the IMS Monitor activity. Restricting the scope has the advantage of reducing the impact of the Monitor on system throughput. However, you should not compromise the collection of useful data. You can control what specific types of events are traced by using specific keywords on the /TRACE command. For example, you can monitor line activity, scheduling and termination events, activity between application programs and message queues, activity between application programs and databases, or all activity. You can also limit monitoring to:

- Particular databases, partitions, or areas
- Particular dependent regions
- A specified interval of time

Obtaining IMS Monitor Reports

You can obtain reports based on the IMS Monitor's output by using the IMS Performance Analyzer (IMS PA) or the IMS Monitor Report Print utility (DFSUTR20). For information on IMS PA, see "Using the IMS Performance Analyzer for z/OS" on page 97.

The IMS Monitor Report Print utility summarizes and formats the raw data (except Fast Path data) produced by IMS and presents the information in a series of reports. You can suppress the reports pertaining to DL/I calls and tabulated frequency distributions.

The duration of the monitored events is determined by the entries for start and stop of the Monitor. You cannot select a different time period for reporting, because many of the timed events are not captured continuously but only when the Monitor is started and stopped. For this reason, you should ensure that the Monitor is stopped before taking any action to stop the IMS control region.

Related Reading: For a description of the JCL requirements and utility control statements, see *IMS Version 9: Utilities Reference: System*.

Using the //DFSSTAT Reports

The //DFSSTAT reports give you the number of database and data communications calls issued by an application program and describe the buffering activity. These reports are described in *IMS Version 9: Utilities Reference: System*.

Using GTF Trace

You can use the MVS Generalized Trace Facility (GTF) to record a wide range of system-level events. The trace activity is controlled from the MVS system console using the MODIFY command. Output is spooled to a sequential data set that is used by a generalized formatting utility. You can write exit routines that are called by the formatting utility to edit the trace records and present the data as desired. The following GTFPARS reports are of special interest for IMS:

- Job Summary report

  This report lists the following for each of the IMS message regions and for the control region: SVC counts, loading of modules, and EXCP, SIO, and IO timings.
- Detail Trace report lists the following:

  For IMS message regions: detailed analysis of program management activity. This analysis gives you the ability to investigate event sequences in detail.

  For the IMS control region: database I/O that is handled in the control region, and network-related activity.

  For all IMS regions running in parallel: Contention between regions (for example, contention for access to a program library), and the flow of control from the control region to the MPPs.
- Job Summary report of Master Address Space in MVS

  This report shows system paging activity.
- Detail Trace report of Master Address Space in MVS

  You can use this report for a detailed investigation of delays caused by paging.
- System Summary report

  You can use this report for I/O subsystem analysis for all IMS data sets.

Using the MVS Component Trace (CTRACE)

IRLM 2.1 uses the MVS component trace (CTRACE) facility to trace IRLM activity. Because the trace output is in MVS CTRACE format, you can use IPCS CTRACE format, merge, and locate routines to process the buffer data.

Use the MVS `TRACE CT` command to start, stop, or modify an IRLM diagnostic trace. This command can be entered only from the MVS master console. Entering the commands requires an appropriate level of MVS authority. IRLM does not support all the options available on the command. You can also start the IRLM tracing by placing `TRACE=YES` in the irlmproc.

Related Reading: For information on the `TRACE CT` command for IRLM, see *IMS Version 9: Command Reference*. For complete information on the command, see *z/OS MVS System Commands*.

See “Tracing IRLM Activity” on page 114 for more information on IRLM traces.

Obtaining Program Isolation and Lock Traces

In an IMS DB/DC or DBCTL environment, you can detect contention for a database segment by examining the output produced by the Program Isolation Trace Report utility (DFSPIRP0). To get the source data for the utility, issue the `/TRACE SET ON PI OPTION ALL` command. To stop gathering source data, issue the `/TRACE SET OFF PI` command. A control statement for the utility can select a start or stop time relative to a specified date.

Tracing the program isolation function creates additional log records. These records contain the enqueue or dequeue requests issued by the program isolation function between sync points as a result of database updates, checkpoint, and message handling events.

The Program Isolation Trace Report utility only reports those events that required wait time. The report identifies the data management block (DMB) name, database control block (DCB) number, relative byte address (RBA), program specification block (PSB) name, and transaction code. The utility sorts all activity by RBA number (shown as ID in the report). The report lists elapsed times for enqueues that required a wait (during the trace interval) and totals the number of enqueues for each ID, DCB, and DMB. The requesting PSB or transaction is considered the holding PSB or transaction of the next enqueue waiting for the same segment. A sample report is illustrated in Figure 6. In this report no elapsed wait time is recorded for Fast Path.

Related Reading: For details of how to run the report utility, see *IMS Version 9: Utilities Reference: Database and Transaction Manager*.

```
                     P R O G R A M   I S O L A T I O N   T R A C E   R E P O R T         PAGE   1
 DATE:  08/10/89
 TIME:  16:36  TO  16:37
 DCB *** REQUESTING *** ELAPSED   **** HOLDING *** ID TOTAL DCB TOTAL DMB      TOTAL
 DMB NAME NUM  ID  TRAN AND PSB NAMES   TIME       TIME    TRAN AND PSB NAMES ENQ'S ENQ'S ENQ'S
TABLEDBQ 1 0022D020  DE1Q     PROGDE1Q  16:36:54   0:00.061  DE2Q     PROGDE2Q
                                                                                  1
           003BE00C  DE2Q     PROGDE2Q  16:36:51   0:00.027  DE1Q     PROGDE1Q
                                                                                  1
           007D901C  DE2Q     PROGDE2Q  16:36:34   0:00.036  DE1Q     PROGDE1Q
                                                                                  1
           008EF014  DE2Q     PROGDE2Q  16:36:49   0:00.038  DE1Q     PROGDE1Q
                                                                                  1
           0090401C  DE1Q     PROGDE1Q  16:36:50   0:00.072  DE2Q     PROGDE2Q
                                                                                  1
           00A06010  DE2Q     PROGDE2Q  16:36:38   0:00.046  DE1Q     PROGDE1Q
                                                                                  1
           00A1401C  DE1Q     PROGDE1Q  16:36:50   0:00.008  DE2Q     PROGDE2Q
                                                                                  1    7    7
TABLEDBR 1 002A901C  DE2R     PROGDE2R  16:36:40   0:00.034  DE1R     PROGDE1R
                                                                                  1
           0045801C  DE2R     PROGDE2R  16:36:41   0:00.043  DE1R     PROGDE1R
                                                                                  1
           0072F024  DE1R     PROGDE1R  16:36:30   0:00.053  DE2R     PROGDE2R
```

Figure 6. Sample Program Isolation Trace Report

You can use the File Select and Formatting Print utility to select and print trace table and PI entries in the log records in the following ways:

- Specify an `OPTION` statement with the `PRINT` parameter and `COND=E` and `EXITR=DFSERA40` keyword parameters. The output is a report containing the program isolation (PI) trace records formatted in sequential order.

  Related Reading: For an example of this report and an explanation of the headings, see *IMS Version 9: Utilities Reference: System*.
- Select only the log records that contain the trace using the IMS Trace Table Record Format and Print Module (DFSERA60). Specify an `OPTION` statement with the `PRINT` parameter and `COND=E` and `EXITR=DFSERA60` keyword parameters. The output is a report containing the PI trace entries, the DL/I trace entries, and the lock trace entries formatted to show these entries in sequential order. For an explanation of the headings, refer to an assembly listing of the macro IDLIVSAM TRACENT.

You can use a Program Isolation Trace Record Format and Print Module output report to find out more information:

- The level of control (LEV) column shows read only, share, exclusive control, and single update activity.

- The return code (RC) column shows return codes from DFSFXC10 or the IRLM. You can determine whether the caller had to wait for the requested resource, or if the transaction caused a deadlock situation.
- The PST post code (PC) column shows the cause of the wait. If the entry is X'60', a deadlock occurred.

You can reduce the number of records examined by specifying an additional `OPTION` statement to the File Select and Formatting Print utility so that only records confirming deadlock are printed.

IMS automatically resolves deadlock situations by using dynamic backout. But the detection of deadlocks is important so you can modify your application design to prevent future deadlocks.

The advantage of the PI trace records report is that it shows where contention for a particular segment or range of segments occurs. The report also shows which transactions are competing within a database. It also shows high wait times that might explain a delay in response time. One way to handle the segment contention might be change the database design to separate some of the fields into an additional segment type.

Using the Trace Facility

You can use the IMS Trace facility to write IMS trace tables internally or to an external trace data set. IMS can write this external trace data set to either DASD or tape:

- DASD data sets can be allocated by JCL or can be dynamically allocated.
- Tape data sets must be dynamically allocated.

Related Reading: For information on how to use the DFSMDA macro to create the dynamic allocation members, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

You can also write the trace tables to the OLDS, but this could adversely affect OLDS performance. The external trace data sets are independent of the OLDS, so you can write trace tables to the external trace data sets even if the OLDS is unavailable.

To display the status of traces, use the `/DISPLAY TRACE` command. This command can be used to determine the status of the IMS traces in effect and the status of any external trace data sets in use.

Related Reading:

- See *IMS Version 9: Diagnosis Guide and Reference* for information about:
  - When and why trace tables are used
  - The DFS2867A message when using external tracing to OLDS
- See *IMS Version 9: Installation Volume 2: System Definition and Tailoring* for information about defining and setting up trace facilities.

Executing Recovery-Related Functions

While IMS is running, an IMS system programmer or operator might need to execute functions relating to the recoverability of the system. These functions include:

- Issuing DBRC commands
- Dumping the message queues
- Recovering the message queues
- Archiving the OLDS
- Making databases recoverable or nonrecoverable
- Using the database recovery service
- Running recovery-related database utilities

Issuing DBRC Commands

You can issue batch and online (/RMxxxxxx) commands and DBRC API requests to use DBRC functions. Authorization checking is supported for DBRC commands that are submitted as batch or online commands. The IMS security administrator can use a security product like RACF (Resource Access Control Facility), a security exit routine, or both to control authorization of DBRC commands. Commands issued through the MVS console are always authorized.

Recommendation: Allow operators to use the /RMLIST and /RMGENJCL commands Restrict the use of /RMCHANGE, /RMDELETE, and /RMNOTIFY commands, because they update the RECON data set.

Related Reading:

- For information on using DBRC commands in a data-sharing environment, see “Online DBRC Commands” on page 217.
- For information on how to control command authorization, see *IMS Version 9: Administration Guide: System*.
- For detailed information about DBRC commands and DBRC API requests, see *IMS Version 9: DBRC Guide and Reference*.

Dumping the Message Queues

If you want to save the message queues in a nonshared-queues environment, use the /CHECKPOINT SNAPQ command. This command dumps the message queues to the log without terminating the online system.

Recommendation: Schedule the /CHECKPOINT SNAPQ regularly because it shortens the time required for emergency restart if a problem occurs on the message queue data sets. Consider the following intervals:

- Whenever the OLDS is switched
- Once each hour
- Once each shift
- Twice each day (midnight and noon)
- Once each day

For a shared-queues environment, use the /CQCHKPT SHAREDQ command to dump the shared queues.

Recovering the Message Queues

In a non-shared-queues environment, you can recover the message queues during an IMS restart if the previous shutdown included the DUMPQ or the SNAPQ keyword. Specify the BUILDQ keyword on the /NRESTART or /ERESTART command to restore the messages to the message queue data sets from the IMS log. Specify the FORMAT keyword on the /NRE or /ERE command if you also want to reinitialize the message queue data sets.

In order to use the /NRE BUILDQ command, the system must be shut down using a /CHECKPOINT DUMPQ | PURGE command. To use the /ERE BUILDQ command, you need only a prior /CHECKPOINT SNAPQ command.

Restriction: If a /NRE BUILDQ or /ERE BUILDQ command fails and you cold start IMS or cold start DC, messages are lost and are not processed.

You can use the Queue Control Facility (QCF) program product (5697-E99) to select messages from the OLDS (or SLDS) and reinsert them into the IMS message queues after an IMS cold start.

Related Reading: See *IMS Queue Control Facility for z/OS* and *IMS Version 9: Diagnosis Guide and Reference* for more information about QCF.

For a shared-queues environment, CQS automatically rebuilds the message queues if the coupling facility fails. You can also use the SETXCF START,REBUILD command to rebuild the queues manually.

Archiving the OLDS

You should archive the OLDS to an SLDS at regular intervals. If you are not using automatic archiving, the MTO should use the DBRC GENJCL command at regular intervals to generate the JCL for the Log Archive utility, and should execute the utility.

Making Databases Recoverable or Nonrecoverable

You can change recoverable full-function DBs to nonrecoverable (after deleting recovery-related records from the RECON data set) by using the CHANGE.DB NONRECOV command. You can change to recoverable again by using the CHANGE.DB RECOVABL command.

Use the LIST.DB command or the DBRC QUERY TYPE DB API request to display whether a database is recoverable.

Running Recovery-Related Utilities

Depending on your recovery strategy, the MTO might be responsible for executing various recovery-related utilities at regular intervals. These could include:

- Database Image Copy utility (DFSUDMP0)
- Online Database Image Copy utility (DFSUICP0)
- Database Change Accumulation utility (DFSUCUM0)

The MTO should also run these utilities when a database changes from nonrecoverable to recoverable.

Modifying and Controlling System Resources

You establish the initial settings of IMS resources during IMS system definition. The MTO, and other operators authorized to do so, can change various system resources using IMS commands.

You can use many IMS commands to perform similar control functions for different types of resources. "List of Commands with Similar Functions for Multiple Resources" shows the relationship between these commands and resources.

Related Reading: For details of these commands, see *IMS Version 9: Command Reference*.

List of Commands with Similar Functions for Multiple Resources

The following tables show the IMS commands that affect certain resources. The resources are:

- Telecommunication Line, Physical Terminal, or Node (Table 9)
- Table 10 (Table 10)
- Table 11 (Table 11)
- Table 12 (Table 12)
- Transaction (Table 13)
- Transaction Class (Table 14)
- Program (Table 15)
- Database (Table 16)
- Area (Table 17)
- Subsystem (Table 18)

Telecommunication Line, Physical Terminal, or Node

Table 9 shows the IMS commands that affect the telecommunication line, physical terminal, or node resources. This table indicates whether each resource can perform the following functions after the command is issued:

- Receive input
- Send output
- Output message queuing

Table 9. IMS Commands That Affect Telecommunications Line, Physical Terminal, or Node Resources

	Resource: Telecommunications Line, Physical Terminal, or Node		
IMS Command	**Receive Input**	**Send Output**	**Output Message Queuing**
/ASSIGN	Y	Y	Y
/LOCK	N	N	Y
/MONITOR	Y	N	Y
/PSTOP	N	N	Y
/PURGE	N	Y	Y
/RSTART	Y	Y	Y
/START	Y	Y	Y

Table 9. IMS Commands That Affect Telecommunications Line, Physical Terminal, or Node Resources (continued)

	Resource: Telecommunications Line, Physical Terminal, or Node		
IMS Command	**Receive Input**	**Send Output**	**Output Message Queuing**
/STOP	N	N	Y
/UNLOCK	Y		Y
Note: /MONITOR, /PSTOP, /PURGE, and /RSTART refer to the telecommunication line or physical terminal, not to the node.			

Logical Terminal

Table 10 shows the IMS commands that affect logical terminal resources. This table indicates whether these resources can perform the following functions after the command is issued:

- Receive input
- Send output
- Queuing from other terminals
- User

Table 10. IMS Commands That Affect Logical Terminal Resources

	Resource: Logical Terminal		
IMS Command	**Receive Input**	**Send Output**	**Queuing from Other Terminals**
/ASSIGN	Y	Y	Y
/LOCK	N	N	N
/PSTOP	N	N	Y
/PURGE	N	Y	N
/RSTART			
/START	Y	Y	Y
/STOP	N	N	N
/UNLOCK		Y	Y

Logical Link Path (MSNAME)

Table 11 on page 107 shows the IMS commands that affect logical link path (MSNAME) resources. This table indicates whether these resources can perform the following functions after the command is issued:

- Queue primary requests to MSNAME
- Send primary requests to MSNAME
- Queue secondary requests to MSNAME
- Send secondary requests to MSNAME
- Queue messages to start a conversation
- Send messages to start a conversation
- Queue messages to continue a conversation
- Send messages to continue a conversation

Table 11. IMS Commands That Affect Logical Link Path Resources

Resource: Logical Link Path (MSNAME)	IMS Command		
	/PURGE	/START	/STOP
Queue primary requests to MSNAME	Y	N	N
Send primary requests to MSNAME	N	Y	N
Queue secondary requests to MSNAME	Y	Y	Y
Send secondary requests to MSNAME	N	Y	N
Queue messages to start a conversation	Y	Y	N
Send messages to start a conversation	N	Y	N
Queue messages to start a conversation	Y	Y	Y
Send messages to continue a conversation	N	Y	N

Logical Link

Table 12 shows the IMS commands that affect logical link resources. This table indicates whether these resources can perform the following functions after the command is issued:

- Receive input
- Send output

Table 12. IMS Commands That Affect Logical Link Resources

IMS Command	Resource: Logical Link	
	Receive Input	Receive Output
/PSTOP	N	N
/RSTART	Y	Y

Transaction

Table 13 shows the IMS commands that affect transaction resources. This table indicates whether these resources can perform the following functions after the command is issued:

- Message scheduling by transaction
- Message queuing by transaction

Table 13. IMS Commands That Affect Transaction Resources

IMS Command	Resource: Transaction	
	Message Scheduling by Transaction	Message Queuing by Transaction
/ASSIGN	Y	Y
/LOCK	N	Y
/MSASSIGN	Y	Y
/PSTOP	N	Y

Table 13. IMS Commands That Affect Transaction Resources (continued)

	Resource: Transaction	
IMS Command	**Message Scheduling by Transaction**	**Message Queuing by Transaction**
/PURGE	Y	N
/START	Y	Y
/STOP	N	N
/UNLOCK	Y	Y
UPDATE	Y	Y

Transaction Class

Table 14 shows the IMS commands that affect transaction class resources. This table indicates whether these resources can perform transaction scheduling by class after the command is issued.

Table 14. IMS Commands That Affect Transaction Class Resources

	Resource: Transaction Class
IMS Command	**Transaction Scheduling by Class**
/ASSIGN	Y
/MSASSIGN	Y
/START	Y
/STOP	N

Program

Table 15 shows the IMS commands that affect program resources. This table indicates whether the program can be started after the command is issued.

Table 15. IMS Commands That Affect Program Resources

	Resource: Program
IMS Command	**Can the program be started?**
/ASSIGN	Y
/LOCK	N
/START	Y
/STOP	N
/UNLOCK	Y

Database

Table 16 shows the IMS commands that affect database resources. This table indicates whether the database can be accessed after the command is issued.

Table 16. IMS Commands That Affect Database Resources

	Resource: Database
IMS Command	**Can the database be accessed?**
/DBDUMP	N

Table 16. IMS Commands That Affect Database Resources (continued)

	Resource: Database
IMS Command	Can the database be accessed?
/DBRECOVERY	N
/LOCK	N
/START	Y
/STOP	N
/UNLOCK	Y
UPDATE	Y or N[1]

[1] Whether the database can be used after the UPDATE command is issued depends on the parameters and keywords specified in this command.
UPDATE DB START(ACCESS) Y
UPDATE DB STOP(ACCESS) N
UPDATE DB STOP(UPDATES) Y
UPDATE DB STOP(SCHD) N
UPDATE DB SET(LOCK(ON)) N	

Area

Table 17 shows the IMS commands that affect area resources. This table indicates whether the area can be accessed after the command is issued.

Table 17. IMS Commands That Affect Area Resources

	Resource: Database
IMS Command	Can the area be accessed?
/DBRECOVERY	N
/START	Y
/STOP	N
UPDATE	Y or N[1]

[1] Whether the database can be used after the UPDATE command is issued depends on the parameters and keywords specified in this command.
UPDATE AREA START(ACCESS)
Y
UPDATE AREA STOP(ACCESS) N
UPDATE AREA STOP(SCHD) N	

Subsystem

Table 18 shows the IMS commands that affect subsystem resources. This table indicates whether the subsystem can be attached after the command is issued.

Table 18. IMS Commands That Affect Subsystem Resources

	Resource: Subsystem
IMS Command	Can the subsystem be attached?
/START	Y
/STOP	Y
/CHANGE	N

You can also use other commands to affect the operating state of specific resources, as described in the following sections.

Related Reading: For more information about using IMS commands, see *IMS Version 9: Command Reference*.

Modifying Dependent Regions

Use the /ASSIGN TRAN or UPDATE TRAN SET(CLASS(*new_class_number*)) command to modify the assignment of classes to regions. Do this to adjust the processing load among message regions.

Modifying Telecommunication Lines

Use the /DEQUEUE command to discard response-mode output messages before you enter an /RSTART LINE command.

Modifying Terminals

Use the /ASSIGN LTERM command to modify the assignment of logical terminals to physical terminals or nodes. The new assignment remains in effect until the next cold start or until you issue another /ASSIGN command.

Use the /DEQUEUE command to discard response-mode output so that the /RSTART command can reset terminal response mode.

Use the /COMPT command for VTAM terminals (nodes) to notify IMS that a terminal component is operable or inoperable.

IMS provides a VTAM I/O Timeout facility to detect VTAM hung nodes and determine what action, if any, should be taken. Use the /TRACE command to start and stop the VTAM I/O Timeout facility. Use the /IDLE command to deactivate a node and the /ACTIVATE command to activate a node. Use the /DISPLAY command to display all nodes that have I/O outstanding for a time period greater than that specified during system definition.

Modifying and Controlling Transactions

Use the /ASSIGN TRAN or UPDATE TRAN SET(CPRI(*new_current_priority*)) command to reassign the scheduling priorities established for transactions during system definition. The new assignments remain in effect until the next cold start or until you issue another /ASSIGN or UPDATE command.

In a shared-queues environment, you can use the /ASSIGN TRAN or UPDATE TRAN SET(CLASS(*new_class_number*)) command to control which IMS subsystems can run certain types of transactions by assigning transactions to particular classes.

Example: You can define TRANA to class 4 on IMSA and to class 255 on IMSB and IMSC, so that only IMSA can run TRANA. If IMSA fails, you can reassign TRANA on either IMSB or IMSC to a class that these IMS subsystems can run.

Recommendation: Do not use the /STOP TRAN or UPDATE TRAN STOP(Q,SCHD) command to control which IMS subsystems can run certain types of transactions.

Controlling Databases

Use the /DBDUMP DB or UPDATE DB STOP(UPDATES) command to stop online update access to a database. This lets you produce an offline dump of the database.

Use the /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command to stop all online access to a database. Use it to recover a database offline.

Normally, IMS switches to using the next OLDS when you enter the /DBDUMP or /DBRECOVERY command. This switch does not occur if you specify the NOFEOV keyword on either command. When you enter the UPDATE DB STOP(ACCESS) command to stop online access to the database, IMS does not switch to the next OLDS. You can specify OPTION(FEOV) if you want IMS to switch to the next OLDS.

Specify the GLOBAL keyword on the /DBDUMP or /DBRECOVERY command to have the command apply to all subsystems sharing the database. The IRLM must be active if you use this keyword. The default is LOCAL, which specifies that the command applies only to the subsystem on which you enter the command.

Modifying Assignments of Intersystem Communication Users (Subpools)

Use the /ASSIGN command to change the assignment of a static LTERM to an ISC user (also called a subpool). The new assignment remains in effect until the next cold start or until you issue another /ASSIGN command.

Modifying ETO User IDs

For dynamic user IDs, use the /ASSIGN command to change the assignment of a user ID to another user or to an LTERM. The new assignment remains in effect until the next cold start or until you issue another /ASSIGN command.

Use the /DISPLAY USER DEADQ command to list all message queues that are eligible for dead letter status. Use the /ASSIGN command to assign a dead letter queue to another user ID. Use the /DEQUEUE command to discard a dead letter queue.

In a shared-queues environment, use the /DISPLAY QCNT MSGAGE command to determine which messages, if any, are eligible for dead letter status.

Modifying Multiple Systems Coupling Resources

Use the /MSVERIFY command to verify the consistency of MSC system identifications (SYSIDs) and logical link paths (MSNAMEs) across two systems. You can use the /MSASSIGN command to change the assignment of MSNAMEs and SYSIDs to logical links.

All changes made by an /MSASSIGN command remain in effect until the next cold start or until you issue another /MSASSIGN command.

After using the /MSASSIGN command, you should use the /MSVERIFY command to ensure that the assignment produced a valid configuration.

Modifying Security Options

Use the /CHANGE command to update a current password with a new password. The current password must be known to IMS.

Restriction: IMS does not allow different user IDs to have the same passwords.

Use the /MODIFY PREPARE RACF and /MODIFY COMMIT commands to reinitialize RACF information if you are not using a RACF data space. If you are using a RACF data space, use the RACF SETROPTS RACLIST command rather than the IMS /MODIFY command.

Use the /DELETE command to delete terminal or password security for the specified system resource.

Use the /SECURE APPC command to control the RACF security level for input from LU 6.2 devices. Use the /DISPLAY APPC command to show the security level that is currently in effect. When IMS starts, the default is full security.

Use the /SECURE OTMA command to control the RACF security level for input from OTMA clients. Use the /DISPLAY OTMA command to show the security level that is currently in effect. When IMS starts, the default is full security.

Displaying and Terminating Conversations

Use the /DISPLAY CONV command to show the status of all conversations, held or active. You can terminate a conversation if necessary with the /EXIT command, but you should only do this after warning the end user.

Modifying and Controlling Subsystems

Use the /CHANGE command to delete an invalid network identifier (NID). If you need to disconnect from a specific subsystem, use the /STOP command. If the /STOP command does not work, use the MVS MODIFY command.

Gathering Performance-Related Data

IMS provides the DB Monitor and the IMS Monitor that gather and format IMS performance-related data and record this data on a statistics log.

The DB Monitor is available to IMS batch systems; it can monitor the activity between application programs and databases. The IMS Monitor is available to IMS online systems. In addition to performing all of the functions of the DB Monitor, the IMS Monitor can track and record information about activities occurring in the IMS control region and data communication activities.

IMS uses the statistics produced by each Monitor to generate reports. The report programs run offline and print reports that summarize and categorize IMS activities.

Related Reading: For additional information about the monitor report programs, see *IMS Version 9: Utilities Reference: Database and Transaction Manager* and *IMS Version 9: Utilities Reference: System*.

Using the DB Monitor

The DB Monitor records performance data during execution of an IMS DB batch system. The DB Monitor can be active during the entire execution of an IMS batch job, or you can stop and restart it from the system console.

Activating and Controlling the DB Monitor

To activate the DB Monitor, specify MON=Y in the PROC statement of the batch job. When you submit the job, IMS uses parameter substitution to update the PARM field of the EXEC statement with a Y in the appropriate position.

To stop the DB Monitor, the system console operator can use the MODIFY *jobname*,STOP command. Message DFS2215A displays on the system console when the Monitor is inactive.

To reactivate the DB Monitor, the console operator can use the MODIFY *jobname*,START command.

Message DFS2216A displays on the console when the Monitor is active again.

Logging the Data from the DB Monitor

IMS records the data produced by the DB Monitor on either the batch log or a separate DB Monitor log defined by the //IMSMON DD statement. Use the //IMSMON DD statement in the batch procedure to control where the data is logged:

- To store the Monitor records on the batch log, either include a //IMSMON DD DUMMY statement or omit the //IMSMON DD statement entirely.
- To store the Monitor records on a separate DB Monitor log, include a valid //IMSMON DD statement.

If, for any reason, IMS cannot open the DB Monitor log data set specified on the //IMSMON DD statement, IMS displays message DFS2217I on the system console. Batch execution continues, but the Monitor is inactive.

If the DB Monitor log device encounters I/O errors, IMS displays message DFS2219I on the system console. Batch execution continues, but the Monitor is inactive.

If you want to stop the Monitor and force an end-of-volume for the DB Monitor log, use the MODIFY *jobname*,STOPEOV command. When you use the STOPEOV keyword, the batch region does not continue executing until the MVS mount request for a new data set is satisfied.

MODIFY Command Errors

If you enter an incorrect job name on the MODIFY command, MVS issues an error message. If you make some other error while entering the MODIFY command, IMS issues message DFS2218I, followed by either message DFS2215A or message DFS2216A.

Using the IMS Monitor

The IMS Monitor records performance-related data during execution of the IMS online subsystem. When a significant event occurs within IMS while the IMS Monitor is active, IMS passes relevant data to it. The IMS Monitor formats a record describing the event, including the time stamps, and logs the event.

Activating and Controlling the IMS Monitor

If you create a DFSDCMON member in the IMS.SDFSRESL data set, you do not need a DD statement in the IMS, DBC, or DCC procedures for the IMS Monitor because IMS dynamically allocates and deallocates the IMS Monitor data set. Otherwise, to activate the IMS Monitor, you must include a DD statement (using IMSMON as the data set name) in the IMS, DBC, or DCC procedures to specify the IMS Monitor log data set. When you include this DD statement, the IMS Monitor becomes available. If the IMS Monitor is available but inactive, processor usage is unaffected.

To start and stop the IMS Monitor, use the /TRACE command.

Monitoring Using the /TRACE Command

In addition to starting and stopping the IMS Monitor, you can specify the types of events to be monitored using the /TRACE command. You can use the /TRACE command to monitor some or all of the following:

- Telecommunication line and logical link activity
- Scheduling and termination events
- Activity between application programs and message queues

- Activity between application programs and databases (full function and Fast Path)

You can also use the `/TRACE` command to limit the monitoring to:

- Particular databases, partitions, or areas
- Particular dependent regions
- A particular interval of time

Logging the Data from the IMS Monitor

If the IMS Monitor log data set is on tape, IMS issues a tape mount request each time you start the IMS Monitor, and IMS rewinds the tape each time you stop the IMS Monitor. If the IMS Monitor log data set is on DASD, IMS uses the same data set each time you start the IMS Monitor. Therefore, you should process the log after you stop the IMS Monitor before restarting it.

Recommendation: Start the IMS Monitor and allow it to run for a period of time, and then stop it to write a “snapshot” of current activity on the IMS Monitor log.

You must stop the IMS Monitor before you take a shutdown checkpoint in order for the report program to produce usable output.

I/O Errors on the IMS Monitor Log Data Set

If a permanent I/O error occurs on the IMS Monitor log data set, IMS stops the IMS Monitor and issues message `DFS2202`. In this situation, you cannot restart the IMS Monitor until you restart IMS because IMS does not close the IMS Monitor log data set until you shut down IMS.

If the problem that caused the error has not been corrected when you restart IMS, you should specify a different volume or unit for the new execution.

Monitoring IRLM

Use the `MODIFY` *irlmproc*`,STATUS` command to obtain information about IRLM activity. This command provides:

- The IMS IDs of IMS subsystems using this IRLM
- The number of locks that are held and waiting for each subsystem on this IRLM
- Identification of this IRLM: its subsystem name and IRLM number

Tracing IRLM Activity

The IMS Monitor does not collect IRLM trace activity. IRLM uses the MVS component trace (CTRACE) facility. Use the `TRACE CT` command to run the following types of sublevel traces:

DBM Trace interactions with the identified DBMS.

EXP Trace any exception condition.

INT Trace member and group events other than normal locking activity.

SLM Trace interactions with the MVS locking component.

XCF Trace all interactions with the MVS cross-system coupling services.

XIT Trace only asynchronous interactions with the MVS locking component.

Related Reading: For a complete description of the MVS `TRACE CT` command for IRLM, see *IMS Version 9: Command Reference*.

Controlling Log Data Set Characteristics

From time to time, you need to tune and modify log data set characteristics, for example, in the following circumstances:

- After monitoring
- After changing your requirements for system availability, integrity, or operator handling

For summaries of actions required for changing log data set design, see Table 19, Table 20 on page 117, and Table 21 on page 118.

Changing Online Log Data Set Characteristics

Because you can restart IMS (warm start or emergency restart) with all input on SLDS, you can reallocate the OLDSs between a shutdown (or failure) and a subsequent restart. To restart IMS using SLDSs as input, you must delete the PRIOLDS and SECOLDS records from the RECON data sets.

Table 19 lists the actions required to change OLDS characteristics.

Table 19. Changing OLDS Characteristics

Modification	Actions Required
BLKSIZE[1]	1. Shut down IMS. 2. Archive all OLDSs. 3. Delete PRIOLDS and SECOLDS records from the RECON data sets, using the `DELETE.LOG` command. 4. Scratch all OLDSs. 5. Reallocate OLDSs with the new BLKSIZE. 6. Verify WADS space allocation. 7. Restart IMS (from SLDS).
Single to Dual[2]	1. Shut down IMS. 2. Archive all OLDSs. 3. Allocate dual OLDSs. 4. Delete the OLDS records from the RECON data sets, using the `DELETE.LOG` command. The primary OLDS records will be deleted. 5. Change OLDSDEF specification to dual. 6. Change IMS startup procedure (`OLDS` DD statements), if required. 7. Compile DFSMDA macros, if required. 8. Modify operating procedures. 9. Restart IMS.
Dual to Single[2]	1. Shut down IMS. 2. Archive all OLDSs. 3. Delete OLDS record from the RECON data sets, using the `DELETE.LOG` command. The primary and secondary OLDS records will be deleted. 4. Delete secondary OLDSs. 5. Change IMS startup procedure. 6. Modify operating procedures. 7. Restart IMS.

Table 19. Changing OLDS Characteristics (continued)

Modification	Actions Required
BUFNO[3]	1. Shut down IMS. 2. Change the OLDSDEF specification for BUFNO. 3. Verify CSA size. 4. Verify WADS space allocation. 5. Restart IMS.
Space, Location, or Allocation[4]	1. Shut down IMS. 2. Archive all OLDSs. 3. Delete PRIOLDS and SECOLDS records in the RECON data sets, using the `DELETE.LOG` command. 4. Scratch and reallocate OLDSs. 5. Restart IMS (from SLDS).

Notes:

1. **Changing OLDS Block Size**: Changing OLDS block size affects WADS space allocation. For information on how to calculate WADS space requirements, see *IMS Version 9: Installation Volume 1: Installation Verification*. To change WADS space allocation, see Table 20 on page 117. All OLDSs must have the same block size.
2. **Changing the Mode**: You must change your OLDSDEF specification in the DFSVSMxx member in IMS.PROCLIB. IMS initialization requires that at least three pairs of OLDSs be available. You must also reconsider data set placement.

   When changing from single to dual OLDS, each data set in a pair of OLDSs must have the same space allocation (number of blocks).

   Changing the mode from single to dual or from dual to single requires changes in the following operating procedures:

   - Skeletal JCL for archive (ARCHJCL member)
   - Skeletal JCL for log recovery (LOGCLJCL member)
   - Batch JCL for Log Recovery utility
   - Batch backout JCL for online transactions and BMPs

   If you warm start the IMS subsystem after changing from single to dual OLDSs, the `/DISPLAY OLDS` command does not show the secondary OLDSs as IN USE until they have been archived once. The command does, however, show dual OLDS logging.
3. **Changing the Number of OLDS Buffers**: Change BUFNO by changing the OLDSDEF specification for BUFNO in the DFSVSMxx member in IMS.PROCLIB.

   When you modify BUFNO, you should consider also modifying the region size for the VSAM common segment area (CSA). The amount of storage fixed for OLDS buffers is [`BUFNO * BUFFERSIZE`].

   WADS space is also affected by BUFNO. For information on how to calculate WADS space requirements, see *IMS Version 9: Installation Volume 1: Installation Verification*. To change WADS space allocation, see Table 20 on page 117.
4. **Changing Space, Location, or Allocation**: For the recommended method of changing the space or location of your OLDS, or for

reallocating an OLDS on the same volume and with the same space, follow the procedure for changing the BLKSIZE in note 1 above.

You can modify space, location, or allocation without shutting down IMS in a non-XRF environment by using the following procedure:

a. `/STOP OLDS` *nn*

b. Archive all OLDSs.

c. Delete PRIOLDS and SECOLDS records in the RECON data sets, using the `DELETE.LOG` command.

d. Scratch and reallocate OLDSs.

e. `/START OLDS` *nn*

Changing Write-Ahead Data Set Characteristics

Table 20 lists the actions required to change WADS characteristics.

Table 20. Changing WADS Characteristics

Modification	Actions Required
Single to Dual[1]	1. Shut down IMS. 2. Allocate new WADS. 3. Define a DFSMDA member. 4. Add DD statement in IMS JCL, if necessary. 5. Code `WADS=D` in IMS JCL. 6. Modify operating procedures. 7. Restart IMS with `FORMAT WADS` keywords.
Dual to Single[1]	1. Shut down IMS. 2. Code `WADS=S` in IMS JCL. 3. Delete DFSMDA member. 4. Remove DD statement in IMS JCL, if necessary. 5. Modify operating procedures. 6. Restart IMS.
Adding Spare	1. Allocate a spare WADS. 2. Update WADSDEF specification in the DFSVSMxx member of IMS.PROCLIB. 3. Define DFSMDA member. 4. Add DD statement in IMS JCL, if necessary. 5. Modify operating procedures. 6. `/START WADS` *n* (or wait until IMS restart).
Removing Spare	1. `/STOP WADS` *n* (and wait for dynamic deallocation). 2. Scratch spare WADS. 3. Update WADSDEF statement in the DFSVSMxx member of IMS.PROCLIB. 4. Remove DD statement in IMS JCL. 5. Modify operating procedures.
Space, Location, or Allocation[2]	1. Shut down IMS. 2. Scratch and reallocate WADSs. 3. Restart IMS with `FORMAT WADS` keywords.

Table 20. Changing WADS Characteristics (continued)

Modification	Actions Required

Notes:

1. **Changing the Mode**: WADS can be dynamically allocated and deallocated.

   To reflect the new mode for WADSs, you must update the following:

   - Skeletal JCL for the Log Recovery utility (LOGCLJCL member).
   - IMS startup procedure (`WADS=` execution parameter).
   - All recovery procedures implemented to recover WADS errors or to close unclosed OLDSs using WADS.
2. **Changing the Space, Location, or Allocation**: All WADS must have the same space allocation (number of tracks) and be on the same type of device.

Changing System Log Data Set Characteristics

Converting from single to dual SLDSs requires modification in the skeletal JCL (ARCHJCL member) and in all your operational procedures using SLDSs. In the operational procedures, consider using the secondary SLDS when you experience errors in the primary one. No modification is required for online processing because IMS dynamically allocates SLDSs.

Changing the BLKSIZE requires modification in the skeletal JCL (ARCHJCL member). All SLDSs required for online processing must have the same BLKSIZE.

Changing RECON Data Sets Characteristics

Table 21 lists the actions required to change RECON data set characteristics.

Table 21. Changing RECON Data Set Characteristics

Modification	Actions Required
Adding Spare[1]	1. Define a spare RECON data set. 2. Compile DFSMDA macro for the spare data set or add DD statement in IMS JCL and batch JCL if you do not use dynamic allocation.
Removing Spare	1. Delete cluster. 2. Delete DFSMDA member in IMS.SDFSRESL or remove DD statement from IMS JCL and batch JCL if you do not use dynamic allocation.
Replacing Active	**Recommendation:** Stop all IMS subsystems and batch jobs accessing the RECON data sets. 1. Define a spare data set with new space or allocate a spare data set at a new location. 2. `CHANGE.RECON REPLACE (RECON`*n*`)` 3. Define a new spare data set. 4. Continue normal processing.
Single to Dual	**Recommendation:** Stop all IMS subsystems and batch jobs accessing the RECON data sets. 1. Define a spare RECON data set. 2. `CHANGE.RECON DUAL` 3. Define a new spare. 4. Continue normal processing.

Table 21. Changing RECON Data Set Characteristics (continued)

Modification	Actions Required
Note: 1. **Adding a Spare RECON Data Set**: The spare data set must be in VSAM CREATE mode.	

Recommendation: For both online and batch, use dynamic allocation for RECON data sets, and run with at least three RECON data sets.

Formatting Newly Initialized Volumes for OLDS

If a newly initialized (or reinitialized, but unformatted) volume is to contain an OLDS, you must format either the volume or the space occupied by the OLDS before the online system uses it. If you do not format the volume, or if the block size of the new OLDS data set is not the same as the existing OLDS data set, you can expect severe performance degradation and excessive device and channel usage until IMS completely fills the OLDS once. This problem is especially noticeable during emergency restart and during XRF tracking and takeover.

You can use any of the following techniques for formatting a volume for an OLDS:

- Copy an existing OLDS (of the same size) into the new OLDS.
- Copy an existing volume into the new volume, scratch the volume table of contents (VTOC), and allocate the new OLDS.
- Use another IMS subsystem to fill the OLDS (turn on all traces to the log and issue checkpoint commands until the OLDS fills).
- Write a program that either writes at least one byte of data into each track on the volume or fills the OLDS with maximum logical record length (LRECL) blocks.

Connecting and Disconnecting Subsystems

Before an IMS subsystem can access databases in an external subsystem, (another program executing in an MVS address space), such as DB2 UDB for z/OS, you must connect the IMS subsystem to this other subsystem. IMS can only connect to another subsystem if that subsystem is identified in the subsystem member in IMS.PROCLIB. Specify the subsystem member name in the `SSM` EXEC parameter. When specified to and accessed by IMS, the subsystem member name cannot be changed without stopping IMS.

Related Reading: For information on the `SSM` EXEC parameter, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

Connections between an IMS subsystem and another subsystem can occur with or without operator intervention, and without regard to which subsystem is available first.

- Automatic connection: When the SSM EXEC parameter is specified, IMS will automatically establish the connection when it processes the parameter.
- Operator-controlled connection: When the SSM EXEC parameter is not specified, the connection will be established when the `/START SUBSYS SSM` command is entered. The `/START SUBSYS SSM` command specifies the subsystem member in IMS.PROCLIB that IMS will use in connecting to subsystems.

Disconnecting IMS from another subsystem is initiated with the `/STOP SUBSYS` command, and normally completes without operator intervention. IMS will quiesce all dependent region external subsystem activity prior to stopping the subsystem.

Chapter 10. Making Online Changes

IMS supports two levels of change for modifying IMS resources online:

- Local online change, which allows you to make online changes to IMS resources for one IMS
- Global online change, which allows you to coordinate online changes to IMS resources across all IMSs in an IMSplex

Related Reading: For a description of what IMS resources can be modified and how, see *IMS Version 9: Administration Guide: System*.

Global online change is not supported in all IMS environments that support local online change. Environments that require the MODSTAT/MODSTAT2 data sets do not support global online change. Use Table 22 to determine if global online change is supported in your environment.

Table 22. IMS Environments That Support Online Change

Environment	Supports Local Online Change?	Supports Global Online Change?
DB/DC	Yes	Yes
DBCTL	Yes	Yes
DBCTL-standby	Yes	No
DCCTL	Yes	Yes
FDR	Yes	No
RSR tracker	Yes	No
XRF active	Yes	Yes
XRF alternate	No	No

This chapter assumes that you are familiar with online change for IMS, and provides guidance for the operator commands used to make online changes.

Commands Applicable to Online Change

When you want to perform an online change, you must make the inactive libraries (those containing the changes to be made) active. Use the following operator commands:

/MODIFY PREPARE
This command prepares an IMS for local online change and specifies which resources to add, change, or delete. When you issue this command, IMS prevents terminals from queuing messages to database programs or transactions that will be changed or deleted.

/DISPLAY MODIFY
This command produces a list of the resources (that are to be changed or deleted) that have work pending. No changes are accepted when work is pending. Use this command to ensure no work is pending before issuing a `/MODIFY COMMIT` or an `INITIATE OLC` command.

/MODIFY COMMIT
This command applies the changes specified in a preceding `/MODIFY PREPARE` command. When this command completes successfully, modifications persist across all IMS restarts.

/MODIFY ABORT
This command aborts local online change on an IMS and resets the status to what it was before you issued a preceding `/MODIFY PREPARE` command. You should use it when a `/MODIFY PREPARE | COMMIT` command is unsuccessful, and you do not want to retry the online change until later.

INITIATE OLC PHASE(PREPARE)
This command prepares all of the IMSs in the IMSplex for global online change by specifying which resources to add, change, or delete.

When running in an environment without RM (`RMENV=N`), this command performs a local online change on the IMS that processes it.

INITIATE OLC PHASE(COMMIT)
This command commits global online change on all IMSs in an IMSplex.

When running in an environment without RM (`RMENV=N`), this command performs a local online change on the IMS that processes it.

TERMINATE OLC
This command aborts global online change on all IMSs in an IMSplex.

When running in an environment without RM (`RMENV=N`), this command performs a local online change on the IMS that processes it.

QUERY MEMBER
This command displays status or attribute information about IMSs in an IMSplex, including local or global online change status, if applicable.

QUERY OLC
This command provides information about the OLCSTAT data set contents.

Local Online Change Command Sequence

When you want to bring changes online locally in one IMS, you must make the inactive libraries (those containing the changes to be made) active. Issue the following sequence of commands:

1. `/MODIFY PREPARE`
2. `/DISPLAY MODIFY`
3. `/MODIFY COMMIT`

You might have to issue additional commands after `/MODIFY PREPARE` to display and resolve work in progress that might cause the commit to fail.

Global Online Change Command Sequence

When you want to bring changes online globally in all IMSs in an IMSplex, you must make the inactive libraries (those containing the changes to be made) active. Issue the following sequence of commands:

1. `INITIATE OLC PHASE(PREPARE)`
2. `/DISPLAY MODIFY`
3. `INITIATE OLC PHASE(COMMIT)`

You might have to issue additional commands after `INITIATE OLC` to display and resolve work in progress that might cause the commit to fail.

When running in an environment without RM (`RMENV=N`), this command sequence performs a local online change on the IMS that processes it.

Changing the System Definition Online

You do not always need to rerun a complete IMS system definition to make changes to the IMS subsystem. Examine the changes to see if an online change can be made.

If the request does not involve changes to the IMS network or to static terminals, you can probably make the change online.

The procedure to make changes to the IMS online system can be summarized:

1. If necessary, make required system definition Stage 1 input changes.
2. If necessary, perform a MODBLKS system definition.
3. If any security authorizations are altered, added, or deleted, run the Security Maintenance utility to prepare the staging library, IMS.MATRIX.

   You can also use the RACF command `SETROPTS RACLIST(`*classname*`) REFRESH` to refresh the RACF resource profiles in the RACF data space. The default resource classes shipped with RACF are listed in *IMS Version 9: Administration Guide: System*, in the section on preparing to use RACF for security.
4. If necessary, run the MFS Language and MFS Service utilities.
5. If necessary, perform DBDGEN, PSBGEN, and ACBGEN.
6. As necessary, use the Online Change Copy utility to produce suitable copies of the IMS.MODBLKS, IMS.ACBLIB, IMS.FORMAT, and IMS.MATRIX data sets.
7. Issue the appropriate online change command sequence to make the online change at the proper time.
8. Verify that the required changes took place.

If you need to undo or back out an online change, repeat the online change command sequence without changing the inactive libraries.

Changing Programs and Transactions

In an IMS DB/DC or DCCTL environment, the online change `PREPARE` command stops all incoming messages and transactions for a resource to be changed, but does not affect messages that have already been queued. The online change `COMMIT` command does not take effect until messages for a resource to be changed or deleted have been scheduled (dequeued), so you must either wait for the messages to be scheduled or stop the associated transactions.

If, for valid reasons, you do not want to wait for the messages to be scheduled (for the online change `COMMIT` command to take effect), use the `/STOP` or `/PSTOP` command to stop the associated transactions before issuing the online change `COMMIT` command.

Recommendation: Plan to make your changes when activity associated with the resource is low—when there are few transactions to be processed. When activity is low, you will not have to wait long for messages to be scheduled or have many transactions to stop if you do not want to wait.

You must stop all message-driven Fast Path programs and wait-for-input programs that refer to resources to be changed or deleted because the online change `COMMIT` command does not execute if any wait-for-input programs have been scheduled.

Also, any active conversations involving resources to be changed or deleted must complete before the online change `COMMIT` command can complete.

Implementing Security Changes Online

While IMS is active, changes to RACF security definitions are made completely outside of IMS. The following general procedure describes how to make changes to RACF:

1. In the RACF database, update the RACF security profiles that describe IMS resources (such as transactions and commands) by issuing RACF commands.
2. Issue the RACF command `SETROPTS RACLIST(classname) REFRESH` to refresh the profiles for the appropriate security class in the RACF data space.

The remaining part of this section describes how to make changes to SMU security while IMS is active.

You can use online change for system definition, including changes to system security. Without restarting the IMS online system, you can:

- Change transaction and terminal authorizations
- Maintain currency of passwords
- Add security for terminals, command transactions, or passwords
- Refresh security matrixes for IMS resource access

The basic procedure is:

1. Make your security definitions ready.
2. Execute the IMS Security Maintenance utility.
3. Coordinate copies of the inactive IMS.MATRIXA/B and IMS.MODBLKSA/B libraries.
4. Give the MTO instructions to perform an online change using the online change `/MODIFY` or `INITIATE` command.

Preparing for Online Security Changes

To modify the security matrixes using an online change, you need to prepare the changes to the Security Maintenance utility input and then:

1. Run the Security Maintenance utility to prepare the staging library, IMS.MATRIX.
2. Use the Online Change utility to produce a suitable copy of the IMS.MATRIX data set in the inactive library.
3. Use the Online Change utility with a `PARM` value of `U` to copy from the staging library to the inactive IMS.MODBLKSA/B library.

   This step is necessary because active MATRIX data sets must correspond to active MODBLKS data sets. When the IMS starts, it determines the active security matrixes by checking the entries in the IMS.MODSTAT data set (or the OLCSTAT data set, if global online change is enabled) for the DD statement of the initially active IMS.MODBLKSA/B data set. For example, if IMS.MODSTAT contained an entry `MODBLKSA`, the data set specified on the `MATRIXA` DD statement would be currently in use for security checking.
4. Tell the MTO when to perform the online change.

If your online security change is only for modifying security checking tables and not for modifying transactions, programs, or databases, you do not need to perform a MODBLKS system definition. However, you must use an unmodified MODBLKS data set when you execute the Security Maintenance utility.

Activating Online Security Changes

The MTO enters a sequence of online change commands to replace the set of security matrixes and resource checking tables. The prevailing security options and timing determine what is replaced:

- If transaction authorization is inactive because the control region JCL specifies the EXEC parameter `TRN=N`, IMS does not load the equivalent security tables as a result of the online change command.
- If you want security maintenance to be done on a specific day or not at a busy time, give these instructions to the MTO.

To begin the online change process, enter the `/MODIFY PREPARE` or `INITIATE OLC PHASE(PREPARE)` command with the `MODBLKS` or `ALL` keyword. If you include a new type of security checking within the inactive security data set, use the `/MODIFY PREPARE MODBLKS` or `INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS)` with the *option* parameter, where *option* can be one or more of the following keywords: `PSWD`, `RACF`, `TERMINAL`, or `TRANCMDS`.

Restriction: The `RACF`, `TERMINAL`, and `TRANCMDS` keywords are not supported for the IMS DBCTL environment.

If you omit an *option* for a type of security checking that is not currently active, IMS does not implement it. If you include an *option* for a type of security checking that is currently active, IMS ignores the *option*. If the content of the inactive security data set conflicts with the currently active security options, IMS rejects the command.

To complete the online change process, enter the online change `COMMIT` command to make the replaced security tables active. The previously used data set, either IMS.MATRIXA or IMS.MATRIXB, now becomes the inactive one. The changes in security checking remain in effect across restarts, or until a subsequent online change replaces the security checking tables.

If a problem arises that prevents the successful completion of the online change, none of the security changes take effect. You should follow up with the MTO to verify that your required security changes take effect.

Changing MFS Formats

In the IMS DB/DC and DCCTL environments, the online change `PREPARE` command does not stop the queuing of related messages when you change MFS formats. You can use the online change `COMMIT` command to change MFS formats without stopping related activity (messages can be queued and scheduled while you change an associated MFS format block). But you should coordinate changing MFS formats and message queuing so that problems do not occur.

Making Online Changes for DEDBs

In the IMS DB/DC and DBCTL environments, you can make both database-level and area-level changes to Fast Path data entry databases (DEDBs). A database-level change affects the structure of the DEDB, and includes such changes as adding or deleting an area, adding a segment type, or changing the randomizer routines. An area-level change involves increasing or decreasing the size of an area, including the overflow portions (IOVF and DOVF) and the control interval (CI) size.

A database-level change requires you to stop all areas of the DEDB. An area-level change requires you to stop the area using either the /DBRECOVERY AREA or UPDATE AREA STOP(ACCESS) command.

IFP and MPP regions that access the changed DEDBs do not need to be stopped during the online change.

Related Reading: For more information about adding, modifying, or deleting DEDBs online, see *IMS Version 9: Administration Guide: Database Manager*.

Making Online Changes in an XRF Complex

In the IMS DB/DC and DCCTL XRF environments, in addition to preparing the staging libraries, you need to prepare the inactive libraries.

If an XRF takeover occurs while the active IMS is processing a /MODIFY COMMIT command, the change might or might not have occurred. At the end of an online change, IMS writes a log record; the XRF alternate subsystem reads the log and makes the online changes to match the changes on the active subsystem. Use the /DISPLAY DB or QUERY DB command to check that a change occurred on the alternate subsystem.

Making Global Online Changes in a Sysplex that Uses XRF

The XRF alternate does not participate directly in a global online change. The alternate participates indirectly once it receives the X'70' log record written by the XRF active system. The QUERY MEMBER command might show some online change status for the XRF alternate while it is processing the X'70' log record.

During takeover of an active system, the XRF alternate updates the OLCSTAT data set with its IMSID. After the takeover is complete, the XRF alternate behaves like any IMS active system and can participate in a global online change. The abended XRF active system can be brought back up as the new XRF alternate, if desired.

The MODSTAT and MODSTAT2 data sets are not used. You can leave the MODSTAT and MODSTAT2 data set DD statements in your IMS control region JCL, but they are ignored.

Global online change in an XRF environment has the following limitations:

- If an XRF active system fails or is shut down (/SWITCH SYSTEM ACTIVE or /CHE FREEZE) and the XRF alternate system takes over, the IMSID for the abended XRF active system continues to be stored in the OLCSTAT data set until you remove it. After takeover, if you bring up the abended XRF active system as the new XRF alternate and initiate the global online change, the XRF alternate might be directed to perform online change phases. In this situation, the XRF alternate system ignores the online change phases and responds that the phases are not applicable.

  If the IMSID for the new XRF alternate is in the OLCSTAT data set, perform one of these steps:

  - When you issue the next INITIATE OLC PHASE(PREPARE) command, specify the FRCABND keyword.
  - Delete the IMSID for the new XRF alternate from the OLCSTAT data set by using the delete function (FUNC=DEL) of the Global Online Change utility DFSUOCL0. Doing this eliminates the need to specify the FRCABND keyword.

- If the XRF active system fails and the XRF alternate system attempts to take over before it catches up with the XRF active system's global online change state, the XRF alternate system abends with U2801 RC=X'0011'. The XRF alternate is not synchronized with the OLCSTAT data set. Because other IMSs in the IMSplex might be synchronized with the OLCSTAT data set, the XRF alternate is not permitted to takeover.

Global Online Change after XRF Takeover and DBCTL Standby Emergency Restart

After an XRF takeover or an emergency restart of a DBCTL standby with RMENV=N and OLC=GLOBAL, two IMSIDs are stored in the OLCSTAT data set (the XRF active and XRF alternate IMSIDs or the DBCTL active and DBCTL standby IMSIDs).

You must run the Global Online Change utility (DFSUOLC0) with FUNC=DEL to delete the active XRF or DBCTL IMSID from the OLCSTAT data set. Otherwise, the next INITIATE OLC PHASE(PREPARE) command you issue to initiate global online change will fail.

Related Reading: For more information about XRF and DBCTL standby, see "IMS DBCTL Capabilities" on page 263.

Online Change Performance Considerations

Deleting databases from MODBLKS using Online Change incurs the performance overhead of reading the intent lists from ACBLIB for all application programs in MODBLKS whose intent lists are not loaded. This is necessary to determine which transactions need to stop queueing, to prevent work that might cause commit to fail. This performance overhead is proportional to the number of application programs defined in MODBLKS for which the intent list is not loaded.

Recommendation: If you need to delete databases from MODBLKS, consider performing an IMS cold start instead of using local or global online change.

Performing a Cold Start

Under certain circumstances—for example, if an IMS goes down during an online change—you might need to perform a cold start. The process you follow to perform a cold start varies depending on whether the IMS uses local or global online change.

For general information about cold start, see "Performing a Cold Start" on page 79.

Local Online Change and Cold Start

Using local online change affects all types of IMS system definitions. IMS online subsystems place control blocks in the IMS.MODBLKS staging library. Before you perform a cold start, copy the MODBLKS library to either the IMS.MODBLKSA or the IMS.MODBLKSB data set. You must also copy the IMS.MATRIX data set.

If you copy these data sets to the inactive library, update the IMS.MODSTAT data set to change the IMS.MODBLKS data set DD name. If you copy to the active library, you cannot, in case of problems, return to the last system environment before the change.

Global Online Change and Cold Start

Typically, an IMS that was down during a global online change needs to cold start when it comes back up. A cold start might be required to prevent restart from processing log records against the current online change libraries, potentially causing restart to fail or resulting in a severe error later.

Before the first IMS in the IMSplex cold starts the first time, you must initialize the OLCSTAT data set. To initialize the OLCSTAT, use the Global Online Change utility (DFSUOLC0).

Important: Use the Global Online Change utility with caution, so you do not inadvertently destroy valid OLCSTAT data set contents. If you do destroy valid OLCSTAT data set contents, global online change and initialization of additional IMSs fail until the OLCSTAT data set is re-initialized.

You should establish an OLCSTAT data set recovery procedure to deal with the loss of the OLCSTAT data set. After every successful global online change, record the modify ID, the active online change library suffixes, and the list of IMSs that are current with the online change libraries. If the OLCSTAT data set is destroyed, run the initialize function of the Global Online Change utility with the saved data to re-initialize the OLCSTAT data set.

Related Reading: For more information about the Global Online Change utility, see *IMS Version 9: Utilities Reference: System*.

Determining When to Perform a Cold Start

Consider these guidelines to determine when to perform a cold start if global online change is enabled:

- Perform a warm start when an IMS was down during the last global online change and the restart does not conflict with the last global online change.
- Perform a cold start when an IMS was down during the last global online change and its restart type conflicts with the last online change type.

  For more information about the types of restart commands permitted for specific online change types, see Table 23.
- Perform a cold start when an IMS was down for two or more global online changes.

Table 23. Restart Commands Permitted for Specific Online Change Types

Online Change Type	Restart Commands Permitted
ACBLIB	/NRE CHECKPOINT 0 /ERE COLDBASE
ALL	/NRE CHECKPOINT 0
FORMAT	/NRE CHECKPOINT /NRE /ERE /ERE COLDCOMM /ERE COLDBASE
MODBLKS	/NRE CHECKPOINT 0

Cleaning Up After a Failed Restart Attempt

If an IMS was down during global online change and your restart command was rejected because the IMS required a cold start, you might need to perform one or more cleanup steps to ensure a successful cold start. Otherwise, the IMS cold start might fail. Perform the following steps to clean up after a failed restart attempt:

- Issue a DBRC `LIST.RECON` command to check for an IMS subsystem record or any open OLDS.
- If a subsystem record exists, issue a DBRC `DELETE.SUBSYS` command to clean up DBRC's subsystem record for that IMS.
- If there are any open OLDS, issue a DBRC `GENJCL.CLOSE` command to enable DBRC to read the RECON and determine which OLDS to close for that IMS.

Making Local Online Changes in a Sysplex

IMS does not automatically coordinate local online changes across a sysplex. Use global online change to coordinate online change across a sysplex. This section describes how to manually coordinate local online changes across a sysplex.

Recommendation: Clone your IMS subsystems and share libraries (ACBLIB, MODBLKS, MATRIX, FORMAT) across the sysplex. MODSTAT IDs must be the same across the sysplex.

To prepare for performing IMS online change in a sysplex environment:

1. Use an AO application program.

   The AO application program should be able to:

   - Stop resources you want to change or delete.
   - Handle online change exception conditions for an individual IMS subsystem.
   - Issue the `/MODIFY PREPARE` or `/MODIFY ABORT` commands.

   Design the AO exit routine to look for the online change messages that IMS issues and to take the suggested action. Design an AO application program or use an automation tool to issue commands to resolve any online change problems.
2. Define the active, inactive, and staging libraries.
3. Share the active and inactive libraries.

   All IMS subsystems that you want to be able to change online should also share the staging libraries.
4. Define each IMS (with its own MODSTAT data set).

   Each IMS must have its own MODSTAT data set, with the same active and inactive ACBLIB, FMTLIB, MATRIX, and MODBLKS definitions.
5. Start each IMS.

To perform the online changes:

1. Perform system definitions on the staging libraries for changes.
2. Run the Online Change Copy utility.
3. Issue `/MODIFY PREPARE` for each IMS subsystem.

   You can have either the IMS operator or an AO application program issue this command. If the command fails on any IMS subsystem in the sysplex, the operator or AO application program must issue the `/MODIFY ABORT` command on all IMS subsystems for which the `/MODIFY PREPARE` succeeded.
4. Stop all resources that will be changed or deleted.

   This step is optional, but it greatly increases the chance that the online change will succeed for an individual IMS subsystem.
5. Resolve any work in progress.

6. Issue /MODIFY ABORT|COMMIT for each IMS subsystem.
7. Start resources that were changed or added.

The /DISPLAY MODIFY command shows only local work in progress, except for status relating to transactions (including TRAN QUEUEING, TRAN SUSPENDED, and TRAN PROCESSING). TRAN QUEUING shows the sum of local and global queue counts.

Making Global Online Changes in an IMSplex

Global online change is an IMS function to change resources for all IMSs in an IMSplex. A Master IMS control region, as specified by the user or by an Operations Manager (OM), uses Resource Manager (RM) to coordinate the phases of online change with other IMSs in an IMSplex.

| If you are running an IMSplex that does not use RM, you can specify the CSL global parameter RMENV=N in the DFSCGxxx IMS.PROCLIB member. If this parameter is specified, IMS does not require RM and will not use RM services, but type-2 command support is available. If OLC=GLOBAL, use the INITIATE OLC command to initiate local online change. The OLCSTAT data set must be defined. This data set cannot be shared between IMS systems.

With global online change, the Master IMS control region coordinates online change of the following resources among all the IMSs in an IMSplex:

- Databases (DMB in ACBLIB)
- Database directories (DDIR in MODBLKS)
- MFS formats (FMTLIB)
- Programs (PSB in ACBLIB)
- Program directories (PDIR in MODBLKS)
- Security matrices
- Transactions (SMBs in MODBLKS)

To enable an IMSplex for global online change, you must:

1. Run the Global Online Change utility (DFSUOLC0) to initialize the OLCSTAT data set.

   The Global Online Change utility needs to be run only once before the first IMS cold starts the first time. The OLCSTAT data set is comparable to the MODSTAT data set used by local online change.

   Important:

   - Use the Global Online Change utility with caution, so you do not inadvertently destroy valid OLCSTAT data set contents. If you do destroy valid OLCSTAT data set contents, global online change and initialization of additional IMSs fail until the OLCSTAT data set is re-initialized.

     You should establish an OLCSTAT data set recovery procedure to deal with the loss of the OLCSTAT data set. After every successful global online change, record the modify ID, the active online change library suffixes, and the list of IMSs that are current with the online change libraries. If the OLCSTAT data set is destroyed, run the initialize function of the Global Online Change utility with the saved data to re-initialize the OLCSTAT data set.

     Related Reading: For more information about the Global Online Change utility, see *IMS Version 9: Utilities Reference: System*.

- For an IMSplex running without an RM (with CSL global parameters `RMENV=N` and `OLC=GLOBAL`), you must ensure that another IMS subsystem does not attempt to use the OLCSTAT data set from the IMS subsystem that is running without RM.

  Related Reading: See *IMS Version 9: Administration Guide: System* for more information.

2. For each IMS in the IMSplex, perform these steps:
   a. Remove the MODSTAT DD and MODSTAT2 DD statements from the IMS control region JCL.
   b. Define DFSCGxxx IMS.PROCLIB member parameters related to global online change.

      Specify OLC=GLOBAL to enable global online change.

      Specify OLCSTAT= with the online change status data set name. All of the IMSs in the IMSplex must specify the same OLCSTAT data set.
   c. Shut down IMS.
   d. Cold start the IMS.

IMS provides several commands to support global online change. The `INITIATE OLC PHASE(PREPARE)` command coordinates the online change prepare phase for all of the IMSs in the IMSplex. Likewise, the `INITIATE OLC PHASE(COMMIT)` command coordinates the online change commit phase and the `TERMINATE OLC` command coordinates the online change abort phase across all the IMSs in the IMSplex. `INITIATE` and `TERMINATE` are similar to the `/MODIFY` command; however, they apply to an IMSplex-wide global online change. The `QUERY MEMBER TYPE(IMS)` command displays status or attribute information about IMSs in the IMSplex.

Initiating a Global Online Change

The `INITIATE OLC` (initiate online change) command is provided to support global online change, where online change is coordinated across IMSs in the IMSplex. `INITIATE OLC` is similar to `/MODIFY PREPARE` and `/MODIFY COMMIT`, except that it applies to an IMSplex-wide global online change. OM sends the `INITIATE OLC` command to one IMS in the IMSplex. For syntax, environment, and usage information, see *IMS Version 9: Command Reference*.

Note: When running in an environment without RM (`RMENV=N`), this command performs a local online change on the IMS that processes it.

`INITIATE OLC` prepares for or commits a global online change of IMS resources across an IMSplex.

`INITIATE OLC` is not supported if local online change is enabled. The `INITIATE OLC PHASE(PREPARE)` command is rejected if the IMS to which the command is routed does not support global online change. If the `INITIATE OLC PHASE(PREPARE)` command is rejected and there is an IMS that supports global online change, route the command to an IMS that supports global online change.

The correct online change command sequence is the following:
1. `INITIATE OLC PHASE(PREPARE)`
2. `INITIATE OLC PHASE(COMMIT)`

Use the /DISPLAY MODIFY command to display the work in progress for resources to be changed or deleted, before attempting the COMMIT phase. When COMMIT is successful, the modifications persist across all IMS restarts, unless global online change occurs while an IMS is down.

If the INITIATE OLC PHASE(PREPARE) is specified without a FRCABND or FRCCNRML keyword and the command fails because one or more IMS are down or go down before the online change is committed, the online change must be aborted and started over. Issue the TERMINATE OLC command to abort the online change.

If the INITIATE OLC PHASE(PREPARE) is specified with the FRCNRML keyword and the command fails for any IMS, you can proceed with an INITIATE OLC PHASE(COMMIT) command, if desired, after shutting down those IMSs where the prepare failed. Otherwise you must abort the online change and start over.

If the INITIATE OLC PHASE(PREPARE) is specified with the FRCABND keyword and the command fails for any IMS, you can proceed with an INITIATE OLC PHASE(COMMIT) command, if desired, after cancelling those IMSs where the prepare failed. Otherwise, abort the online change and start over.

If the INITIATE OLC PHASE(COMMIT) command fails for any IMS before the OLCSTAT data set is updated, you can either correct the errors and try the commit again or abort the online change with a TERMINATE OLC command.

If the INITIATE OLC PHASE(COMMIT) command fails for any IMS after the OLCSTAT data set has been updated, you can correct the errors and try the commit again. The online change cannot be aborted.

If an IMS abends during online change and the INITIATE OLC PHASE(PREPARE) command was not specified with FRCABND, then issue the TERMINATE OLC command to abort the online change. The INITIATE OLC PHASE(COMMIT) command is not permitted in this case. If an IMS abends during online change and the INITIATE OLC PHASE(PREPARE) command was specified with FRCABND, then the INITIATE OLC PHASE(COMMIT) command is permitted.

Type-1 commands and type-2 commands that come from the OM interface are rejected during the commit phase, if the command changes resources. Commands that change resources could interfere with the online change of the resources. Type-1 commands and type-2 commands that come from the OM interface are permitted during the commit phase, if the command displays resources, such as /DISPLAY or QUERY. Type-1 commands that are entered from the system console or an IMS terminal are queued during the online change commit phase. These commands run after the online change is committed or aborted.

You can specify INITIATE only through the OM API.

INITIATE is invalid on an XRF alternate, RSR tracker, and FDR system.

Each IMS participating in the global online change does not issue the same synchronous online change messages to the master terminal or system console that it does for a local online change. The OM command response contains information equivalent to the online change messages that appear for the local online change, such as the DFS3499 message contents.

Each IMS participating in the global online change may issue asynchronous online change messages to the system console, such as DFS3400, DFS3445, and DFS3498. For more information about these messages, see *IMS Version 9: Messages and Codes, Volume 2*.

The OM command time-out default of 300 seconds (5 minutes) might not be enough time for the online change phase to complete. You might need to specify a time-out value on the command based on the needs of the installation.

INITIATE OLC Return and Reason Codes

An IMS return and reason code is returned to OM by the `INITIATE OLC` command. The IMS return and reason codes are defined in DFSCMDRR.

The `INITIATE OLC` command master usually performs the online change phase locally first. If the online change phase fails locally, the command master usually skips sending the online change phase to the other IMSs, sets a completion code for each other IMS indicating that the online change phase was not attempted, and terminates command processing. However, if the `INITIATE OLC PHASE(COMMIT)` command fails on the local IMS because of work in progress, the command master still sends the commit phase 1 to the other IMSs. The purpose is to report work in progress for all the IMSs in the IMSplex, to facilitate completion of work in progress.

In a mixed IMSplex, you might have some flavors of IMS that support a particular online change flavor and some that do not. At least one IMS in the IMSplex must successfully perform the online change phase for the command to be considered successful. If no IMS in the IMSplex supports the online change flavor, the command reason code indicates that none of the IMSs performed the online change phase. If you enter an `INITIATE OLC PHASE(PREPARE)` command flavor that does not apply to any IMS in the IMSplex, you must terminate the online change with a `TERMINATE OLC` command.

Related Reading: For a list of return and reason codes that might be returned by the `INITIATE OLC` command to OM, see *IMS Version 9: Messages and Codes, Volume 1*.

Handling Errors for the INITIATE OLC Command

The `INITIATE OLC` command can result in errors that leave one or more of the IMSs in the IMSplex in various online change states. Correct the error. Issue the `QUERY MEMBER` command and the `QUERY OLC` command to help you decide whether to terminate the online change by issuing `TERMINATE OLC` or to try the `INITIATE OLC` command again.

Before attempting a global online change, issue the `QUERY OLC LIBRARY(OLCSTAT) SHOW(MODID)` command to get the current modify ID. If the `INITIATE OLC` command fails, issue the `QUERY OLC LIBRARY(OLCSTAT) SHOW(MODID)` command again, to see if the modify ID is the same. If the modify ID increased by 1, the online change is considered to be successfully completed.

If the `INITIATE OLC` command fails, issue the `QUERY MEMBER TYPE(IMS) SHOW(STATUS)` command to display the online change state of all the IMSs in the IMSplex. Evaluate the `QUERY MEMBER TYPE(IMS) SHOW(STATUS)` output to help you to determine what to do. For information about the `QUERY MEMBER TYPE(IMS) SHOW(STATUS)` output and the action to take, see “Displaying Status or Attribute Information about IMSs in the IMSplex” on page 134.

Displaying Status or Attribute Information about IMSs in the IMSplex

The `QUERY MEMBER` command displays status or attribute information about IMSs in the IMSplex, including online change status. You can specify `QUERY MEMBER` only through the OM API. For syntax, environment, and usage information, see *IMS Version 9: Command Reference*.

Displaying Information about the OLCSTAT Data Set

The `QUERY OLC LIBRARY` command provides information about the OLCSTAT data set contents. For syntax, environment, and usage information, see *IMS Version 9: Command Reference*.

Terminating an Online Change

The `TERMINATE OLC` command supports aborting a global online change across an IMSplex. The `TERMINATE OLC` command can also be shortened to `TERM OLC`.

A `TERMINATE OLC` command that aborts a global online change is similar to the `/MODIFY ABORT` command, except that a global command applies to all IMSs in an IMSplex that are participating in the global online change.

OM sends the `TERMINATE OLC` command to an IMS in the IMSplex.

You can use `TERMINATE OLC` to abort an IMSplex-wide global online change initiated by a `INITIATE OLC PHASE(PREPARE)`, before the online change is successfully committed with a `INITIATE OLC PHASE(COMMIT)`.

Use `TERMINATE OLC` to abort an online change after an `INITIATE OLC PHASE(COMMIT)` failure that occurs before the OLCSTAT data set is updated. After the commit process has updated the OLCSTAT data set, the online change is considered to be successful and cannot be aborted.

`TERMINATE OLC` is not supported if local online change is enabled. The `TERMINATE OLC` command is rejected if the IMS to which the command is routed does not support global online change. If this occurs and there is an IMS that supports global online change, the user must route the command to a specific IMS that supports global online change.

You can specify `TERMINATE OLC` only through the OM API.

The `TERMINATE OLC` command can be processed by DB/DC, DBCTL, and DCCTL environments.

The `TERMINATE OLC` command is invalid on the XRF alternate, RSR tracker, and FDR system.

The OM command time-out default of 300 seconds (5 minutes) might not be enough time for the online change phase to complete. You might need to specify a time-out value on the command based on the needs of the installation.

The command syntax for this command is defined in XML and is available to automation programs which communicate with OM. For more information about command syntax, environments, and usage, see *IMS Version 9: Command Reference*.

Handling Errors for Global Online Change Commands

When a global online change command results in an error, the action you perform depends on the state of the IMSs in the IMSplex after the online change error has occurred. Table 24 describes actions that you can perform when an online change command results in an error.

Table 24. Handling Errors for Online Change Commands

IMSplex State after Online Change Command Error	User Action to Perform
One or more IMSs in prepare complete state after a prepare failure.	Issue the TERMINATE OLC command. Correct the problem before reissuing the command.
All IMSs in prepare complete state.	Issue the INIT OLC PHASE(COMMIT) or TERMINATE OLC command.
A mix of IMSs in prepare complete and commit phase 1 complete state.	Correct the problem and try the commit again or issue the TERMINATE OLC command.
All IMSs in commit phase 1 complete state before OLCSTAT data set updated.	Correct the problem and try the commit again or issue the TERMINATE OLC command.
All IMSs in commit phase 1 complete state after OLCSTAT data set updated.	Correct the problem and try the commit again. The online change is committed and cannot be aborted.
Mix of IMSs in commit phase 1 and commit phase 2 complete state.	Correct the problem and try the commit again.
All IMSs in commit phase 2 complete state.	Correct the problem and try the commit again.
Mix of IMSs in commit phase 2 complete state and not in online change state.	Correct the problem and try the commit again.

Global Online Change Scenarios

The following scenarios describe events that can occur during the processing of online change commands and explains how the IMSs respond to the commands based on their current state.

Scenario 1: A Successful Global Online Change

This scenario shows the output returned to the TSO SPOC as the result of the following successful global online change command sequence:

1. The QUERY OLC LIBRARY(OLCSTAT) command is issued to show the active online change libraries, ACBLIBA, FMTLIBA, and MODBLKSA. The command also shows that the current modify ID is 1 and the OLCSTAT data set name is IMSTESTL.IMS.OLCSTAT. The IMSs that are current with the online change libraries include SYS3, IMS2, and IMS3. These IMSs will be permitted to perform a warm start or emergency restart.
2. The INIT OLC PHASE(PREPARE) TYPE(MODBLKS) OPTION(FRCABND,FRCNRML) is issued to prepare IMSs for a MODBLKS online change. IMS3 is the prepare master because it builds all of the output lines. Because the FRCABND and FRCNRML options are specified when the online change command is issued, any IMSs that are down are permitted to perform the online change. IMS2, SYS3, and IMS3 are in the OLCSTAT data set, so they will participate in the online change.
3. The INIT OLC PHASE(COMMIT) is issued to commit the online change. IMS2, IMS3, and SYS3 all successfully complete the online change. The commit master, IMS3, reports the new online change libraries, MODBLKSB, ACBLIBA, and FMTLIBA, and the new modify ID of 2.

4. Another QRY OLC LIBRARY(OLCSTAT) SHOW(ALL) command is issued to show the new online change libraries and the new modify ID of 2.

   The following sample output represents 4 responses to the 4 commands described in this scenario:

```
Response for: QRY OLC LIBRARY(OLCSTAT) SHOW(ALL)
MbrName  CC  Library  ACBLIB  FMTLIB  MODBLKS  Modid  DSName        LastOLC    MbrList
IMS3     0   OLCSTAT    A     A       A        01     IMSTESTL.IMS.OLCSTAT     SYS3,IMS2,IMS3

Response for: INIT OLC PHASE(PREPARE) TYPE(MODBLKS) OPTION(FRCABND,FRCNRML)
MbrName  Member     CC ACBLIB FMTLIB MODBLKS    Modid
IMS3     IMS2       0
IMS3     SYS3       0
IMS3     IMS3       0       A      A      A        1

Response for: INIT OLC PHASE(COMMIT)
MbrName  Member     CC ACBLIB FMTLIB MODBLKS    Modid
IMS3     IMS2       0
IMS3     SYS3       0
IMS3     IMS3       0      A      A       B         2

Response for: QRY OLC LIBRARY(OLCSTAT) SHOW(ALL)
MbrName  CC  Library  ACBLIB  FMTLIB  MODBLKS  Modid  DSName        LastOLC    MbrList
IMS3      0   OLCSTAT    A        A        B       02  IMSTESTL.IMS.OLCSTAT  MODBLKS  SYS3,IMS2,IMS3
```

Scenario 2: Online Change Prepare Command is Rejected because an IMS is Down

In this online change scenario, the INITIATE OLC PHASE(PREPARE) command is rejected because two IMSs are down, but no FRCABND or FRCNRML options were specified when the command was issued. Table 25 identifies the command sequence and codes returned during this scenario:

Table 25. Scenario 1: Command Sequence and Codes for Failed Online Change Prepare Command

Action or Event	IMS A Completion Code	IMS B Completion Code	IMS C Completion Code	Command Return Code	Command Reason Code
IMS B abends					
IMS C shuts down normally					
INIT OLC PHASE(PREPARE) TYPE(ALL)	2	B2	B2	C	3004
INIT OLC PHASE(PREPARE) TYPE(ALL) OPTION(FRCABND,FRCNRML)	0	B2	B2	4	100C
INIT OLC PHASE(COMMIT)	0	B2	B2	4	100C
INIT OLC PHASE(PREPARE) TYPE(ALL)	0			0	0
INIT OLC PHASE(COMMIT)	0			0	0

1. An INITIATE OLC PHASE(PREPARE) TYPE(ALL) command was issued. IMS A is the command master. IMS A detects that IMS B and IMS C are down and that no FRCABND or FRCNRML keywords were specified, so it rejects the command. IMS A returns a completion code of 2 (not done) for itself; IMS A also returns a completion code of B2 (IMS state error) for IMS B and IMS C. IMS A sets an overall return code of C and an overall reason code of 3004 (failed for all IMSs).
2. Another INITIATE OLC PHASE(PREPARE) TYPE(ALL) command was issued, this time with FRCABND and FRCNRML. These options force the online change even though IMSs are down. IMS A is again the command master. IMS A detects that IMS B and IMS C are down, but determines that this is acceptable. IMS A sets a completion code of B2 (IMS state error) for IMS B and IMS C. IMS B

and IMS C do not perform the prepare phase, because they are down. IMS A performs the prepare phase locally and returns a completion code of 0 for itself. IMS A then sets an overall return code of 4 and a reason code of 100C (failed for some IMSs for acceptable reasons).

3. An INIT OLC PHASE(COMMIT) is issued; the commit succeeds. IMS A updates the OLCSTAT data set with the new information. IMS A includes a record for itself, but omits a record for IMS B and IMS C. IMS B and IMS C are no longer current with the online change libraries. IMS B and IMS C will have to cold start because the last online change included MODBLKS. If an IMS is down during a MODBLKS online change, it has to cold start.
4. Another INITIATE OLC PHASE(PREPARE) command is issued. This time, IMS B and IMS C do not participate because they are not current with the online change libraries. Only IMS A participates in the global online change.
5. An INIT OLC PAHSE(COMMIT) command is issued. IMS B and IMS C do not participate because they are not current with the online change libraries. Only IMS A commits the global online change. IMS A sets a completion code of 0 for itself and an overall return code and reason code of 0.

Scenario 3: Commit Phase 1 Fails Due to Work in Progress

In this online change scenario, commit phase 1 fails due to work in progress. However, the INITIATE OLC PHASE(COMMIT) command eventually succeeds on all of the IMSs, which results in overall return and reason codes of 0. Table 26 identifies the command sequence and codes returned during this scenario:

Table 26. Scenario 2: Command Sequence and Codes for Failed Commit Phase 1

Action or Event	IMS A Completion Code	IMS B Completion Code	IMS C Completion Code	Command Return Code	Command Reason Code
INIT OLC PHASE(PREPARE) TYPE(ALL)	0	0	0	0	0
INIT OLC PHASE(COMMIT)	B1	B1	4	C	3004
INIT OLC PHASE(COMMIT)	B1	4	4	C	3004
/DISPLAY MODIFY and other commands	none	none	none	none	none
INIT OLC PHASE(COMMIT)	0	0	0	0	0

1. The INITIATE OLC PHASE(PREPARE) command is issued and succeeds on all of the IMSs. IMS A is the command master. Each IMS performs the prepare phase locally and sets a completion code of 0. IMS A sets an overall return code and reason codes of 0.
2. The INITIATE OLC PHASE(COMMIT) command is issued, and IMS A is the command master. IMS A attempts commit phase 1 locally, but the commit phase fails due to work in progress. IMS A sets a completion code of B1 (resource state) for itself. IMS A continues processing and sends the commit phase 1 to the other IMSs that need to report any work in progress. IMS C successfully completes commit phase 1, but IMS B has work in progress. IMS A sets a completion code of 4 (incomplete) for IMS C and quits the commit process at this point due to errors.
3. Another INITIATE OLC PHASE(COMMIT) command is issued; IMS A, is the master again. IMS A again attempts commit phase 1 locally, which fails due to work in progress. IMS A sends commit phase 1 to IMS B and IMS C. IMS B successfully completes commit phase 1. IMS C returns a completion code indicating that it is already in the correct state. IMS A quits the commit due to

errors and sets the completion code to 4 (incomplete) for IMS B and IMS C. IMS A sets an overall return code of C and a reason code of 3004 (failed for all IMSs).

4. /DISPLAY MODIFY and other commands are issued to resolve the work in progress that might cause the commit to fail.
5. A final INITIATE OLC PHASE(COMMIT) command is issued. All of the IMSs successfully complete the commit phases 1, 2, and 3 and set completion codes of 0 for themselves. The overall return code and reason code is 0. IMS A sets an overall return code and reason code of 0.

Scenario 4: Online Change Prepare Command Fails Due to a Time-out Error

In this scenario, the online change prepare command fails due to a time-out error. Because one IMS never receives the prepare phase, the command times out while waiting for a response. Online change must be aborted after a prepare failure. Table 27 identifies the command sequence and codes returned during this scenario:

Table 27. Scenario 3: Command Sequence and Codes for Failed Prepare Command Due to Time-out Error

Action or Event	IMS A Completion Code	IMS B Completion Code	IMS C Completion Code	Command Return Code	Command Reason Code
INIT OLC PHASE(PREPARE) TYPE(MODBLKS)	0	0	91	C	3000
INIT OLC PHASE(PREPARE) TYPE(MODBLKS)				10	4110
INIT OLC PHASE(COMMIT)	2	2	B2	C	3004
TERMINATE OLC	0	0	3	4	100C
INIT OLC PHASE(PREPARE) TYPE(MODBLKS)	0	0	0	0	0
INIT OLC PHASE(COMMIT)	0	0	0	0	0

1. In this scenario, IMS A is the command master, and the INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS) command is issued; IMS A is the command master. SCI goes down before IMS C receives the prepare phase. IMS A and IMS B successfully complete the prepare phase locally and set completion codes of 0 for themselves. The RM process step response times out, and RM returns the process step results to IMS A. IMS A sets a completion code of 91 (time-out) for IMS C. IMS A rejects the prepare command because of the time-out error. After the failure, IMS A and IMS B remain in an online change prepare state, and IMS C is not in an online change state. IMS A sets an overall return code of C and reason code of 3000 (prepare worked for some of the IMSs). Because IMS C does not do the prepare and a prepare failure occurred, the online change must be aborted.

   In this situation, you should issue QUERY MEMBER TYPE(IMS) to see the state of the IMSs to determine what to do.
2. Another INIT OLC PHASE(PREPARE) TYPE(MODBLKS) command is issued. However, because online change must be aborted after a prepare failure, this prepare command also fails. IMS A, the command master, detects that it is already in a prepare state and rejects the command with a return code of 10 and a reason code of 4110. (The INIT OLC command is rejected because the command does not apply to the online change state of the command master.)
3. An INIT OLC PHASE(COMMIT) command is issued. The installation mistakenly tries to commit the online change after the prepare failure. IMS A is once again

the command master. IMS A performs commit phase 1 locally, then sends commit phase 1 to IMS B and IMS C. IMS A and IMS B successfully perform commit phase 1 locally. IMS C rejects the commit phase 1 because it is not in an online change prepare state. IMS C returns a completion code of `B2` (IMS state error). IMS A detects the error and sets the completion code to `2` (incomplete) for IMS A and IMS B. IMS A also returns a completion code of `B2` (IMS state error). IMS A then terminates commit processing. The command fails with a return code of C and a reason code of 3004 (no IMSs successfully completed command).

4. A `TERMINATE OLC` command is issued to abort the global online change. IMS A and IMS B successfully complete the abort. Because IMS C is not in an online change state, it returns a completion code of 3 (IMS already in requested online change state). IMS A, the command master, treats this as acceptable, and returns an overall return code of 4 and a reason code of 100C (successful, but not applicable to some IMSs).
5. An `INIT OLC PHASE(PREPARE) TYPE(MODBLKS)` command is issued. The prepare succeeds.
6. An `INIT OLC PHASE(COMMIT)` command is issued. The commit succeeds.

Chapter 11. Shutting Down IMS

A common sequence for shutting down the entire online system is:

1. For an IMS DB/DC or DCCTL environment, stop data communications. For an IMS DBCTL environment, disconnect from the CCTL.
2. Stop dependent regions.
3. Stop the control region.
4. For an IMS DB/DC or DBCTL environment, stop the IRLM.
5. For a shared-queues environment, if CQS has not shut down, shut it down.
6. For an IMSplex system using CSL, shut down the CSL manager address spaces (OM, RM, and SCI).

The command used to shut down the control region also forces termination of data communications and the dependent regions if they have not already been terminated in an orderly way, and can also tell the CQS subsystem to shut down.

Stopping Data Communications

You can use the `/STOP DC` command to prevent new users from logging on to VTAM terminals. This command does not terminate existing sessions. You can use the `/STOP LINE` and `IDLE LINE` commands to disable BTAM terminals. You can use the `/PURGE TRAN` or `UPDATE TRAN START(SCHD) STOP(Q)` command to prevent existing BTAM and VTAM terminal users from entering new transactions.

To terminate a VTAM session, you can use one of the following commands:

- `/STOP NODE`
- `/CLSDST NODE`

  Recommendation: Use the `/CLSDST NODE` command to terminate a VTAM session instead of the `/STOP NODE` command. The `/STOP` command requires that you subsequently issue a `/START` command before you or a user can initiate a session. You can initiate a session using the `/OPNDST` command.
- VTAM `VARY`
- `/IDLE NODE`

  You can use this command if you already entered a `/STOP DC` or `/CHECKPOINT` command.

Or you can wait for the end user to terminate the session.

To shut down VTAM, use the `HALT NET` command (abbreviated as `Z NET`). If you issue a `Z NET` command with no keywords, you terminate VTAM, and IMS permits no new logons, but existing sessions continue. The existing VTAM sessions must be ended by the terminal user, the VTAM operator, or the MTO. If you issue the `Z NET,QUICK` command, VTAM does not allow new sessions to be established or any additional input or output operations to take place, and IMS terminates existing sessions.

Stopping APPC

Use the `/STOP APPC` command to prevent new users or programs from allocating APPC conversations with the IMS APPC BASE LU. The `/STOP APPC CANCEL` command prevents new users or programs from allocating APPC conversations with any IMS APPC LU. In both cases, the conversation is rejected with a `TP_Not_Available_No_Retry` return code by APPC/MVS.

Stopping OTMA

Use the /STOP TMEMBER command to suspend communications using a specified transaction pipe. If you want to shut down OTMA completely, use the /STOP OTMA command.

Disconnecting from the CCTL

For information about disconnecting from the CCTL, see your CCTL documentation.

Stopping Dependent Regions

You can use the /STOP REGION command to terminate dependent regions. You also terminate dependent regions when you shut down the control region using a /CHECKPOINT command.

Stopping the Control Region

To shut down the IMS control region, use the /CHECKPOINT FREEZE|DUMPQ|PURGE command. You should provide guidelines for the MTO on selecting the right kind of shutdown. Using the FREEZE keyword causes the quickest shutdown, and the PURGE keyword causes the slowest.

- /CHECKPOINT FREEZE|DUMPQ|PURGE immediately terminates sessions for all logical units as follows:

  FREEZE Immediately after current input/output message

  DUMPQ After blocks have been written to the log

  PURGE After all queues are empty
- /CHECKPOINT [FREEZE|DUMPQ|PURGE] QUIESCE allows all network nodes to complete normal processing before IMS shuts down.

Table 28 shows what IMS does in response to each type of checkpoint command.

Table 28. System Actions during Checkpoint

System Action	FREEZE	DUMPQ[1]	PURGE
Stop terminal input	X	X	X
Process transactions		X[2]	X
Free message regions	X	X	X
Purge real storage message queue to disk	X	X	
Log checkpoint data	X	X	X
Notify the CCTL	X		X
Wait for the CCTLs to disconnect from their resource managers	X		X
Write MSDBs to checkpoint data set	X	X	X
Write CHKPT ID to master terminal	X	X	X
Purge database buffer pool	X	X	X
Terminate message regions	X	X	X
Send queued output	X	X	X
Wait for BMP's sync point	X	X	

Table 28. System Actions during Checkpoint (continued)

System Action	FREEZE	DUMPQ[1]	PURGE
Wait for BMP's completion			X
Send shutdown message to terminals		X	X
Dump message queues to log		X	
Close databases	X	X	X
Close queues	X	X	X
Close log	X	X	X
Terminate IMS control region	X	X	X

Notes:

1. The /CHECKPOINT DUMPQ command is not valid for an IMS DBCTL environment.
2. Applies to Fast Path messages only.

Recommendations:

- Have the MTO broadcast a message to all users about 10 minutes before he or she enters the IMS shutdown command. This lets end users, except those involved in conversations, terminate their work in an orderly way.
- If you use the IMS monitor, stop it with the /TRACE SET OFF MON command before using the /CHECKPOINT command.
- Allow active VTAM terminals to complete processing before the IMS shutdown begins. See "Terminating Sessions" on page 146 below.

After the shutdown process has begun, you can use the /DISPLAY SHUTDOWN STATUS command to list the following:

- Communication lines and terminals that still contain active messages
- BMPs and CCTL threads that are still active

To speed up the shutdown process, you can use the /IDLE command to stop I/O operations on specified lines.

In the IMS DBCTL environment, IMS notifies the CCTL that IMS is shutting down. In response, the CCTL must disconnect from the IMS DBCTL subsystem. The CCTL can control how long its threads process before it disconnects, for example, it could let all threads reach a sync point first.

If IMS fails to shut down, and if logging resources are available, you must force IMS to terminate (see "Forcing Termination of IMS" on page 148).

Exceptions:

- If no logging resources are available, the OLDS must be archived before you can force shutdown.
- IMS does not shut down if the CQS subsystem is unavailable while you are running in a shared-queues environment.

Shutting Down IMS Using the /CHECKPOINT Commands

The way IMS shuts down depends on which form of the /CHECKPOINT command you use. The forms of this command are:

- /CHECKPOINT FREEZE

- /CHECKPOINT DUMPQ
- /CHECKPOINT PURGE

Shutting Down IMS Using /CHECKPOINT FREEZE

The fastest way to terminate IMS in an orderly way is to use the /CHECKPOINT FREEZE command. Using the /CHECKPOINT FREEZE command, you allow IMS to complete input and output messages that are in transit. Then IMS:

- Stops each communication line
- Terminates VTAM sessions
- Terminates message regions as soon as the messages currently being processed have completed
- Returns status code XD to batch message regions when they next issue a DL/I checkpoint call

In a DBCTL environment, IMS notifies the CCTL of the /CHECKPOINT FREEZE command so the CCTL can disconnect from its connected resource managers. The CCTL controls how long its CCTL threads can continue processing before the disconnect occurs. For example, the CCTL might let all its threads reach a sync point.

During shutdown, IMS also closes message queue data sets and databases, and writes checkpoint data to the OLDS, just as for a simple checkpoint. IMS issues message DFS994, which contains the last checkpoint ID, and closes the OLDS. IMS updates the checkpoint ID table to reflect the shutdown, and writes it to the RDS. Then IMS closes the RDS and terminates the IMS control region.

During normal shutdown, IMS tells CQS to shut down. If you do not want CQS to shut down, specify the NOCQSSHUT keyword on the /CHECKPOINT command. If you are shutting down a single IMS subsystem in a sysplex in order to activate a system generation, change a parameter specification, or for reasons that are not related to the CQS subsystem, you do not need to shut down the CQS address space. By leaving the CQS address space active after IMS has shut down, CQS can continue to participate in structure checkpoint, overflow, and rebuild events.

Recommendations:

- Ensure that the CQS address space is active during structure checkpoint processing. During a structure checkpoint, MVS deletes log records that are no longer needed for structure recovery, thus reclaiming log space. If CQS is not active during the structure checkpoint, it is possible that the CQS restart log records will be deleted. In this case, the next restart for the CQS address space must be a cold start. You must respond COLD to WTOR message CQS0032A. If you choose to leave CQS up, you can reduce the number of times that CQS needs to cold start due to log record deletion.

  If CQS is active when IMS is started, IMS re-connects to the existing CQS and does not need to wait for CQS initialization and restart to complete.
- Although using the /CHECKPOINT FREEZE command is the fastest way to accomplish an orderly shutdown, you should not use it regularly for normal system termination. When you use the /CHECKPOINT FREEZE command, end users might not receive responses for extended periods of time (until you restart IMS). IMS also rejects Fast Path input messages during shutdown. If during restart, IMS must load the message queues from the log to the message queue data sets (a BUILDQ), you might need to access previously archived OLDSs (to isolate the last DUMPQ or SNAPQ checkpoint).

Shutting Down IMS Using /CHECKPOINT DUMPQ

The /CHECKPOINT DUMPQ command requests an immediate shutdown. Using the /CHECKPOINT DUMPQ command, you allow IMS DB/DC or IMS DCCTL to complete input and output messages that are in transit. Then IMS:

- Stops each communication line
- Terminates VTAM sessions
- Terminates message regions as soon as the messages currently being processed have completed
- Allows Fast Path message-driven regions to process all received messages before terminating
- Returns status code XD to batch message regions when they next issue a DL/I checkpoint call

In a nonshared-queues environment, IMS writes the contents of the message queues to the system log along with checkpoint data. In a shared-queues environment, IMS does not dump the shared queues during shutdown.

The cost of executing a /CHECKPOINT DUMPQ command over a /CHECKPOINT FREEZE command is the time it takes to dump the queues; this time varies depending on the system design, transaction volumes, and activity at a specific time of day.

Shutting Down IMS Using /CHECKPOINT PURGE

Using the /CHECKPOINT PURGE command is the most time-consuming method of terminating IMS. As the result of the /CHECKPOINT PURGE command, IMS stops each input communication line as soon as it receives any messages that are in transit. IMS then:

- Processes all messages in the input queue, if possible (if the transaction and program are not stopped)
- Transmits all output, if possible (if the line and terminal are not stopped)
- Stops output communication lines after it has terminated all active regions and has sent all possible output messages
- Terminates VTAM sessions

IMS writes any unprocessed input messages and any untransmitted output messages to the system log along with checkpoint data.

For an IMS DBCTL environment, the /CHECKPOINT PURGE command operates exactly like the /CHECKPOINT FREEZE command, except that the IMS DBCTL system waits for BMPs to complete their processing.

Forcing an Acceleration of the Shutdown Process

All three variations of full shutdown (/CHECKPOINT FREEZE | DUMPQ | PURGE) allow transactions that had already been started to complete, and allow BMP programs to run to their next checkpoint. These commands might also allow CCTL threads that had already been started to complete. In some cases, you might have to wait a long time before IMS finally shuts down. For example, a conversational transaction or CCTL might be waiting for an end user to enter a response; or a long BMP program might not take any checkpoints.

If necessary, you can force shutdown (although some work might be lost). You can use various forms of the /DISPLAY or QUERY command to discover what work is still running. Then, you can use other commands to force shutdown (for example, the /EXIT, /IDLE LINE, or /STOP REGION commands).

If all else fails, you can bring down IMS by issuing an MVS `MODIFY IMS,DUMP` command or `MODIFY IMS,STOP` command.

Note: Only use the MVS `MODIFY IMS` commands as a last resort, because you must immediately enter an emergency restart (`/ERESTART`) command and shut down again before you can run batch jobs.

Terminating Sessions

Terminating a session releases the terminal from its current logical connection to IMS (the VTAM application program). Terminating a session also makes the terminal available for sessions with other VTAM application programs, or you can terminate communications altogether. Sessions can be terminated by the IMS MTO, the VTAM network operator, or the user at the terminal.

Definitions: There are two types of session termination: orderly and immediate.

Orderly termination
The terminal is allowed to complete normal processing before the session is terminated.

Immediate termination
Forces the terminal to terminate the session unconditionally.

Because these two session termination methods have different consequences, you must develop specific procedures for different situations.

Orderly Termination

To initiate an orderly termination of the network, use the IMS `/CHECKPOINT [FREEZE | PURGE | DUMPQ] QUIESCE` command. When you specify the `QUIESCE` keyword, IMS sends a VTAM shutdown command to all terminals and waits until they all complete normal processing.

After all terminals indicate that shutdown is complete, IMS issues VTAM `CLSDST` macro instruction to cause VTAM to send the `UNBIND` command to all terminals. This command releases the terminals from their sessions with IMS, and prohibits additional data transmission.

During an orderly termination, the IMS MTO can terminate the network unconditionally, rather than wait for the orderly termination to complete, by initiating an immediate termination.

Immediate Termination

To initiate an immediate termination of the network, use the IMS `/CHECKPOINT FREEZE | PURGE | DUMPQ` command. When you enter this command, without the `QUIESCE` keyword, IMS issues the VTAM `CLSDST` macro instruction to cause VTAM to send the `UNBIND` command to all terminals. This command releases the terminals from their sessions with IMS, and prohibits additional data transmission.

You can use the `/CLSDST` or `/STOP` commands to terminate stations or portions of a network selectively. These commands cause IMS to issue the VTAM `CLSDST` macro instruction for the specified stations. The `/STOP` command also prevents additional sessions from being established until you issue a `/START` command for the terminal.

The VTAM network operator can use the `VARY` command to terminate a session immediately. It is sometimes necessary for the VTAM network operator to use this command to terminate sessions that have an error preventing an I/O operation from completing.

The terminal user can use VTAM session control commands to terminate a session with IMS. For normal processing, when the user decides to terminate his or her session with IMS, the terminal should send the VTAM request-shutdown command. IMS completes any input or output currently in progress for that terminal and then issues the VTAM `CLSDST` macro instruction.

If the terminal detects an error condition from which it cannot recover, it can send the VTAM terminate-session command. VTAM releases the terminal from the session and notifies IMS.

Shutting Down an IMS Network

You can shut down the IMS network while shutting down IMS or without shutting down IMS. Use the `/CHECKPOINT` command to terminate the network and shut down IMS. Use the `/STOP DC` command to terminate the network only. You can also use the VTAM `HALT NET` command to shut down the VTAM network.

If your network includes APPC, be aware of the following:

- Allocated LU 6.2 conversations might prolong the shutdown process.
- Network shutdown does not prevent the application programs from establishing LU 6.2 conversations.
- LU 6.2-originated transactions, not scheduled but on the IMS queue, terminate with message DFS1970.

Terminating an ISC Session from CICS

You can only use CICS control operator commands to terminate ISC sessions. The CICS operator can release a session by using the `CEMT SET TERMINAL(`*termid*`) RELEASED | OUTSERVICE` command, where *termid* is the four-character session name (defined on the `DEFINE SESSIONS` command in the CSD utility), or the `TRMIDNT` in the `DFHTCT TYPE=TERMINAL` program. The effects of the keywords for the `CEMT SET TERMINAL` command are described below:

RELEASED

The session terminates when active transactions complete (unless you also specify the `PURGE` or `FORCEPURGE` keywords) and the session is left in a VTAM between-brackets state.

When you specify `RELEASED`, CICS initiates an orderly termination between IMS and CICS. Although, from the CICS point of view, the session may appear to be in warm-start state, it is actually in cold-start state as a result of the VTAM SBI/BIS (Stop Bracket Initiation / Bracket Initiation Stopped) command flow.

OUTSERVICE

The session terminates when active transactions complete (unless you also specify the `PURGE` or `FORCEPURGE` keywords), the session is left in a VTAM between-brackets state, and no additional transactions can use the terminal. For a VTAM terminal, using this keyword also causes it to be released and the operator to be signed off, either immediately or when the current transaction has completed.

When you specify `OUTSERVICE`, CICS requests resynchronization when the session is re-initiated, that is, the session is left in a warm-start state. To initiate a session in cold-start mode, the CICS operator must use the `CEMT SET TERMINAL (`*termid*`) COLDACQ` command (see “Connecting ISC Sessions from CICS to IMS” on page 85 for information about initiating a session).

You can specify both RELEASED and OUTSERVICE together on the same command.

Restriction: The OUTSERVICE keyword is not applicable for LU 6.2 devices.

Recommendation: Although a CICS application program can issue an EXEC CICS DISCONNECT call to initiate orderly session termination, this approach is not recommended in a normal application. However, you can write an “operator control” application to issue this call.

Any messages about the session’s termination are sent to transient data destination CSMT.

Stopping the IRLM

You can stop the IRLM from the system console using one of the following commands:

```
MODIFY irlmproc,ABEND,NODUMP
STOP irlmproc
```

Recommendation: Terminate the IRLM subsystem only after all IMS subsystems connected to the IRLM have completed their processing.

Stopping the CQS

During normal shutdown in a shared-queues environment, IMS tells CQS to shut down, but if it does not, you can use the MVS STOP command to shut down the CQS subsystem.

Normally, you cannot shut down CQS while IMS is running and connected to it, but if you need to shut down CQS in this case, use the MVS CANCEL command. The IMS subsystem does not have access to the shared queues as long as CQS is down, and messages that are locked by that IMS subsystem are unavailable to all other IMS subsystems in the sysplex.

Stopping the CSL Manager Address Spaces

If you are running IMS as an IMSplex with CSL, and you are shutting down the entire IMSplex, you should stop the CSL manager address spaces (OM, RM, and SCI) after you shut down the IMS control region. Note that if you are planning to restart the IMS control region, you do not need to stop the CSL address spaces. The new IMS control region will connect to the existing CSL address spaces when it initializes.

Related Reading: See *IMS Version 9: Common Service Layer Guide and Reference* for information about stopping the CSL manager address spaces.

Forcing Termination of IMS

Usually, you use the /CHECKPOINT command to shut down IMS. However, in certain error situations, such as a control region loop, you might need to force termination of IMS. In this case, use the MVS MODIFY command, and be sure to request a dump of the IMS control region.

You can also use the MVS CANCEL command, specifying the DUMP keyword, to terminate IMS. If you need to enter this command more than once, give each CANCEL time to complete before entering it again.

Attention: Using the MVS CANCEL command more than once might stop IMS from completing termination tasks in the proper order and from releasing MVS system resources. If IMS does not complete termination, you might need to IPL your system before restarting IMS.

Related Reading: See *MVS/ESA System Commands* for more information on the MVS MODIFY and CANCEL commands.

Offline Dump Formatter

Because online dump formatting increases the time required for an online IMS subsystem to terminate, offline dump formatting can reduce the duration of IMS outages for end users. You should be able to restart online subsystems faster as long as MVS remains operational.

Any DL/I batch job can also use offline dump formatting after calling for a SYSMDUMP.

Producing a Dump for Offline Formatting

IMS can produce a machine-readable dump for later offline formatting. This can be an SDUMP, SYSMDUMP, standalone dump, a dump produced using the DUMP command, or any other machine-readable dump, such as an MVS SVC dump.

Producing a Dump Using the MVS MODIFY Command

Recommendation: Use the MVS MODIFY command to create a machine-readable dump rather than an MVS or JES CANCEL command. The CANCEL command often produces undesirable online IMS dump formatting for canceled address spaces in addition to the requested dump for offline formatting.

Producing a Dump Using the MVS DUMP Command

The MVS operator can request a console dump at any time, whether or not IMS is terminating, by entering the following command:

```
DUMP COMM=description
```

MVS responds with the following message:

```
xx IEE094D  SPECIFY OPERAND(S) FOR DUMP COMMAND
```

To which the MVS operator must then reply with the following command:

```
xx,JOBNAME=(imsname,dbrcname,dliname,irlmname),
   SDATA=(PSA,NUC,SQA,RGN,CSA,TRT)
```

Unless you dump all specified areas, the IMS Offline Dump Formatter might not be able to format all desired IMS subsets of the dump.

Producing a Dump Using the Stand-Alone Dump (SADMP)

If MVS fails, the MVS operator can take a stand-alone dump of the entire system. This includes:

- Suspending operation of the system.
- Setting the address of the device containing your High-speed SADMP program, which you have generated to dump real and virtual areas.

  To dump virtual areas, code PROMPT on the AMDSADMP macro.

- Performing a STORE STATUS.
- Initiating the IPL of the stand-alone dump program. This program prompts you for:

```
AMD001A TAPE=____
AMD011A TITLE=_____..._____
```

- Answering the prompts for dumping virtual storage areas:

```
AMDO59D ENTER DUMP OPTIONS, 'LIST', OR 'END'.
> dump CSA,ASID('IMSjobname','DL/Iname','DBRCname','IRLMname')
AMDO59D ENTER DUMP OPTIONS, 'LIST', OR 'END'.
> end
AMD010I PROCESSING ASID=  ASCB= JOBNAME=*MASTER*
```

- Waiting until the stand-alone dump program completes. The following messages signal completion:

```
AMD005I  REAL DUMP DONE
AMD025I  VIRTUAL DUMPING COMPLETE FOR CSA
AMD010I  PROCESSING ASID=0001
   ⋮
AMD023I  VIRTUAL DUMP COMPLETE - 00
```

Recommendation: Do not press the EXTERNAL INTERRUPT key before the dump completes because this could prevent the IMS Offline Dump Formatter from producing a formatted dump later.

Keeping Dump Data Sets Available

To avoid losing dumps of the IMS subsystem, MVS console and IMS master terminal operators must keep the system dump data sets available. MVS operators should transfer and process IMS-produced SDUMPs using the IEBGENER program according to your installation procedures. If you produce IMS dumps using SYSMDUMP DD statements, the MTO should transfer and process the dump data sets according to your installation procedures.

The MVS operators should be aware of the following IMS messages that relate to the offline dump process:

```
DFS3906I  DFSDUMP FAILED BECAUSE ALL SYSTEM DUMP DATA SETS ARE FULL
DFS3906A  REPLY "S" TO SKIP, OR "U" TO RETRY AFTER CLEARING
          A DUMP DATASET.
```

In this case, if a dump is required, reply U after clearing one or more dump data sets. If a dump is not required, reply S.

```
DFS3907I  DFSSUMP FAILED BECAUSE A DUMP IS IN PROGRESS
DFS3907A  REPLY "S" TO SKIP, OR "U" TO RETRY AFTER THE CURRENT
          DUMP COMPLETES.
```

In this case, if the dump is required, reply U after you receive the following message:

```
IEA911E  COMPLETE/PARTIAL DUMP ON SYS1.DUMPxx.
```

If a dump is not required, reply S.

```
IEA793A  NO DUMP DATASETS AVAILABLE FOR DUMP=nnn BY JOB (imsproc).
USE THE DUMPPDS COMMAND OR REPLAY D TO DELELTE THE DUMP
```

In this case, the dump is written to the dump dataset after clearing one or more dump datasets. If a dump is not required, reply D.

For more information about these messages, see *IMS Version 9: Messages and Codes, Volume 1*.

Database Resource Adapter Storage

When the database resource adapter (DRA) is used to access an IMS DB subsystem, the DRA code actually runs in the CCTL region or OS/390 application region, not in any of the IMS DB regions. A shutdown of the IMS DB subsystem with a dump does not produce any information about the DRA.

Chapter 12. Recovering from IMS Failures

When a failure occurs, you must perform the correct recovery and restart procedures to ensure system integrity.

The IMS control region Extended Specified Task Abnormal Exit (ESTAE) routines clean up resources after an IMS abend. The reason for the abnormal termination and the results of cleanup processing have a great effect on the subsequent restart process. The processing sequence of these routines is as follows:

- The storage management task ESTAE routine writes recovery data to the RDS and then closes the RDS.
- The IMS control task ESTAE routine terminates all dependent regions and issues message DFS629I.
- All ESTAE routines attempt to purge the OLDS log buffer and to write a termination log record.
- The ESTAE routines tell DBRC to sign off abnormally.
- The ESTAE routines tell IRLM to quit, and either retain or release locks, based on whether DBRC indicates that databases have been updated.
- The ESTAE routines disconnect XRF alternate subsystems from the MVS availability manager to perform I/O prevention.

In addition, the IMS resource cleanup module cleans up resources whenever IMS terminates. It releases the common storage area (CSA), closes DBDSs, deletes the subsystem interface control block, closes the VTAM Access-method Control Block (ACB), and issues message `DFS627I`.

If you use the DL/I address space (you specified `LSO=S` in the IMS procedure), IMS issues message `DFS603I` to indicate that cleanup is successful.

This chapter describes ways of recovering from the following types of failures:

- MVS system failures
- IMS Control Region Failures
- Emergency Restart Failures
- System Data Set Failures
- Log Errors
- Dependent Region Failures
- Database Failures
- Network Failures
- CPI Communications Failures
- MSC VTAM Message Resynchronization and Recovery
- CQS Failures and Structure Failures
- CCTL Failures
- DBCTL Failures
- IRLM Failures
- User Access Problems

Related Reading: For more information about failure recovery, see *IMS Version 9: Diagnosis Guide and Reference*.

MVS System Failures

If IMS terminates because an MVS failure or a hardware or power failure, you must IPL MVS and then restart IMS using an /ERESTART command.

Database I/O errors, occurring as a result of, or at the same time as, a hardware failure, can leave damaged data on DASD. If IMS terminates abnormally after this kind of an I/O error before IMS can record the I/O error on the IMS log and in DBRC, IMS does not know that there is damaged data. And if IMS does not know that there is damaged data, your data integrity is compromised.

If you experience or suspect I/O errors that lead to these kinds of failures so that it is necessary to use the /ERE OVERRIDE command, check and, if necessary, recover all databases allocated for dynamic backout or DEDB Redo during emergency restart.

If APPC/MVS fails, restart APPC/MVS and issue the IMS /START APPC command. to establish the interface between IMS and APPC/MVS.

Control Region Failures

The following can cause control region failures: software errors causing an abend or a loop, or failures relating to system data sets, including the log data sets. After determining and correcting the cause of the failure, you can restart the system using the /ERE command.

If the failure results in a loop in the control region, you might need to force termination of the region with the MVS MODIFY command, as described in “Forcing Termination of IMS” on page 148.

When the IMS control region fails, APPC/MVS terminates LU 6.2 conversations with the DEALLOCATE abend.

When the IMS control region fails, the CQS subsystem to which it is connected does not shut down. If you need to shut down the CQS subsystem, use the MVS STOP or CANCEL command.

Emergency Restart Failures

Normally, emergency restart backs out incomplete changes from programs that were active at the time of failure. However, damaged logs or message queues might prevent emergency restart from completing. If this occurs, you might be able to perform an emergency restart from a previous checkpoint or by specifying FORMAT ALL on the /ERESTART command.

If you are running an IMS DB/DC system, and an emergency restart fails, you do not have to cold start the entire system. You can use the commands /ERE COLDCOMM and /ERE COLDBASE. These commands cold start part of the system while restarting the other part: /ERE COLDCOMM cold starts the DC part and restarts the rest, while /ERE COLDBASE cold starts the DB part and restarts the rest.

If you are running an IMS DBCTL or DCCTL system, and an emergency restart fails, you must cold start the entire system using the /ERE COLDSYS command.

If no emergency restart succeeds, an appropriate response depends on your understanding the reason for the failure. Although it might not work exactly as written for your installation, use the following procedure to reestablish database integrity:

1. Run the Log Recovery utility (DFSULTR0) to close the OLDS (CLS mode). If the utility needs additional logs, concatenate them in your JCL, oldest first, and rerun the utility.
2. Be sure that your log is correct and that your databases are consistent with the log.
3. Produce an SLDS using the Log Archive utility (DFSUARC0).
4. Forward recover all updated DEDBs to make them consistent with the log using the Database Recovery utility (DFSURDB0).

   Recommendation: If the recovered area utilized the shared VSO option, ensure that there are no failed-persistent XES connections to the CF structure used by the area. If necessary, use the MVS `SETXCF` command to delete the connections prior to cold start of the systems involved.
5. For full-function databases, run the Batch Backout utility (DFSBBO00) to back out programs (PSBs) that were active at the time of failure. Determine these from the output of the Log Recovery utility.
6. Update the RECON data set, if necessary, to reflect the current status of the system.
7. If you use MSDBs, rebuild the MSDBINIT data set from the last used MSDBCP1 and MSDBCP2 data sets and the last SLDS.

At this point, database integrity should be reestablished. If you are using DBRC or data sharing, additional actions might be required before you perform a cold start.

System Data Set Failures

If an IMS system data set fails because of an unrecoverable I/O error, you must scratch, reallocate, and reformat the data set before you restart the system using the `/ERESTART` command.

Message Queue Data Set Failures

This section does not apply to a DBCTL environment or to a shared-queues environment.

Two types of problems can occur with message queue data sets: they can run out of space, or an I/O error can occur. In either case, you need to scratch and reallocate the problem data set, and increase its size if necessary. IMS automatically reconstructs the contents of the message queues when you restart IMS and specify the `BUILDQ` and `FORMAT` keywords on the restart command.

Restriction: IMS can only reconstruct the message queue data sets from the initial cold start or from a SNAPQ or DUMPQ checkpoint.

To recover message queues during IMS restart, use the `/NRE BUILDQ` or `/ERE BUILDQ` command. If you also want to reinitialize the message queue data sets (that is, reformat them with null records), use the `FORMAT` keyword on the restart command.

To use the /NRE BUILDQ command, you must have previously shut down IMS using a /CHECKPOINT DUMPQ | PURGE command. You can use the/ERE BUILDQ command to recover messages from a prior SNAPQ checkpoint (/CHECKPOINT SNAPQ).

If an emergency restart fails, you must initiate an emergency cold start using the /ERE COLDSYS command. An emergency cold start does not perform any database recovery, so you must close and archive the last OLDS.

If an IMS restart fails, you can use the Queue Control Facility (QCF) program product (5697-E99) to recover the message queues if you use one of the following cold start commands after the failed restart: /ERE CHECKPOINT 0, /ERE COLDCOMM, or /ERE COLDSYS. With QCF, you can select messages from the OLDS or an SLDS and requeue them to the message queues after IMS restarts (cold start). QCF provides recovery modes that analyze and select the messages to be requeued. For more information about QCF, see the *IMS Queue Control Facility for z/OS*.

Other System Data Set Failures

Recommendation: Make periodic backup copies of IMS system data sets.

If an error occurs on an IMS data set, such as IMS.ACBLIBx or IMS.FORMATx, you need to recover it. If you have made periodic backup copies of the data set, you can use the latest copy. If you have not changed the content of the data set since making the copy, you can use it as is. If you have changed the content of the data set since making the copy, for example, you have added ACB generations since copying the ACBLIB data set, you need to redo the changes before the backup copy can be considered up-to-date.

If you have not made backup copies of the data set, you need to recreate it. For example, if you lose the active format library (IMS.FORMATx) and you have no backup copy of it, you need to rerun Message Format Services and redo all your format definitions.

If you have inactive data sets from an online change, you can recover those data sets involved with online change (MODBLKS, MATRIX, ACBLIB, and FORMAT). In this case, you must reapply any changes made to the system data sets.

IMS does not offer any specific utilities or commands to recover system data sets.

Recovering RECON

If an I/O error occurs on a RECON data set, DBRC tries to use a spare data set. If one is available, DBRC copies the good RECON data set to the spare, and then activates the spare. When other subsystems using the failed RECON data set complete their processing, you can delete and redefine the error data set so it can be used as a spare. If, however, you want to analyze the RECON error, you should allocate new space for the RECON data set, rather than deleting and redefining it.

If DBRC cannot locate a spare data set, all currently executing jobs will continue processing using the RECON data set in single mode. DBRC allows new jobs to start with only one RECON data set if you specify the STARTNEW keyword on the INIT.RECON or CHANGE.RECON command.

Attention: Do not allow jobs to start with only one RECON data set because it jeopardizes the integrity of the system.

What to Do If Both RECONs Are Unusable

It is unlikely that both RECON data sets would be unusable; however, if both RECON data sets ever become unusable, do the following:

1. Stop all jobs that require access to the RECON data set.
2. If you can access both RECON data sets, use the VSAM access method services `REPRO` command to back them up. This step is optional, but recommended.
3. Use the VSAM access method services utility to delete and redefine your RECON data sets.
4. Use the access method services REPRO command to restore one of the RECON data sets.
5. Use the `REPRO` command to restore the other RECON data set from the first.
6. Use the DBRC `LIST.RECON` command to list one of the RECON data sets. Based on the list, determine which DBDSs IMS updated since you made the backup in step 2. If you cannot determine which DBDSs have been updated, assume that all have been updated.
7. Use the DBRC `CHANGE.IC` command with the `INVALID` keyword to mark all image copy records in error for all applicable DBDSs in step 6.
8. Make an image copy of all applicable DBDSs in step 6.
9. Use the DBRC `BACKUP.RECON` command to make a backup copy of the RECON data sets.

The RECON data sets are now restored and resynchronized with the databases.

If DBRC does not manage a large number of databases, it might be easier to:

1. Stop all jobs that require access to the RECON data set.
2. Define new RECON data sets.
3. Initialize these RECON data sets.
4. Register the environment (always keep a backup copy of the most recently initialized, but not yet used, RECON data set available).
5. Take image copies of all databases.

Finally, before you proceed with regular work, clean up the new RECON data set by, for example, closing any open, out-of-date OLDSs using the `NOTIFY.PRILOG` command.

Log Errors

Three types of errors can occur during logging: A write error on the OLDS, a read error on the OLDS or SLDS, and a read or write error on the WADS.

OLDS write error

When an OLDS has a write error, IMS stops the OLDS (or a pair of OLDSs) in error. No specific MTO interaction is required. When an OLDS is no longer needed for dynamic backout, IMS dynamically deallocates a stopped OLDS.

If you are using single logging, IMS switches to the next OLDS. If the error causes only two OLDSs to remain available, IMS terminates. If an error occurs and no other OLDS is available, IMS abends with code `U0616`.

If you are using dual logging and an error occurs (either to one or both volumes in the pair), IMS switches to the next OLDS pair and continues. What happens next depends on what is specified on the DEGRADE control statement of the DFSVSMxx member.

DEGRADE=YES
Specifies that IMS will continue to use all good pairs of OLDSs until only two pairs remain. At this point, IMS switches (degrades) to single-logging mode, using whatever good data sets are left from each pair of OLDSs. When only two good OLDSs remain, IMS terminates (as it does in normal single-logging mode). DEGRADE=YES is the default.

DEGRADE=NO
Specifies that IMS will terminate when only two good pair of OLDSs remain.

Recommendation: Use the /START OLDS command to allocate new pairs of OLDSs.

OLDS or SLDS read error
When read errors occur on the OLDS or the SLDS during emergency restart, IMS abends. When read errors occur on the OLDS during dynamic backout, backout fails.

WADS error
When a WADS has an I/O error, IMS switches to another WADS if one is available. If none is available, processing continues without a WADS. IMS maintains log-write-ahead protocols by truncating log buffers. As with OLDS errors, no specific MTO interaction is required. You should scratch and reallocate any WADS with a write error after you shut down IMS.

Recovering a Log

Problems can occur with any of the log data sets: OLDS, SLDS, WADS, RDS, and RLDS. Your actions depend on the log, the type of error, whether single or dual logging was in effect when the error occurred, and what processing was being performed when the error occurred. “Using the Log Recovery Utility (DFSULTR0)” describes situations that require using the Log Recovery utility.

| **Related Reading:** For information about Knowledge-Based Log Analysis (KBLA), see *IMS Version 9: Utilities Reference: System*. |

Using the Log Recovery Utility (DFSULTR0)

The Log Recovery utility produces a usable log data set from an OLDS or SLDS that contains read errors or was not properly closed. The Log Recovery utility has three modes of operation:

- CLS mode, to close an OLDS
- DUP mode, to create:

  An interim log containing error records

  A closed batch SLDS containing an end-of-file mark
- REP mode, to do the following:
  - Read the interim log
  - Replace the error records with data that you specify
  - Create a new log

When IMS opens the input logs, DBRC validates them, ensuring that the data set name and volume serial numbers agree with those in the RECON data set. For log recovery, DBRC requires that the entire log data set be specified as input to the IMS Log Recovery utility, not just selected volumes. For a closed SLDS, DBRC requires only the last volume as input to the Log Recovery utility.

When IMS opens the output logs, it calls DBRC to do the following:

- Create an IPRIOLDS record (and ISECOLDS record if you are using dual logging) using the time stamp of the input log and the data set name from the job file control block of the output log. Applies to DUP mode.
- Create new OLDSs from interim OLDSs. Applies to REP mode.

DBRC issues the `GENJCL.CLOSE` command to generate a job to run the Log Recovery utility. When you issue this command, specify the subsystem ID of the IMS subsystem that created the OLDS to be closed. You can also specify the OLDS to be closed. If you do not specify an OLDS, DBRC closes the most recent, open OLDS. You can issue the `GENJCL.CLOSE` command using the Recovery Control utility or as an IMS command.

Because operation of these modes differs for OLDSs and SLDSs, separate sections below describe them in detail.

OLDS Recovery

You must close an OLDS before you can archive it or use it as input to any utility. IMS automatically closes the OLDS during normal shutdown or during emergency restart. You must close the OLDS using the Log Recovery utility in either of the following circumstances:

- When an emergency restart fails and, rather than performing another emergency restart, you perform a cold start
- When IMS does not close the OLDS because IMS detected a write error (single logging only)

The Log Recovery utility recovers the OLDS from the following types of errors:

- An I/O error while reading the input log data set
- An error in the log record
- A sequence error in the log record, the log block, or the OLDS write time stamp

To recover the OLDS, the Log Recovery utility does the following:

- In CLS mode, the utility closes an input OLDS from information in the WADS if IMS cannot close it because of a system failure. Otherwise, the utility closes the OLDS from the OLDS used immediately after the OLDS in error. The utility uses the immediately prior OLDS, if any, to establish a base point for close processing, using the last block sequence number.

  If dual logging is in effect, IMS must close both OLDSs. If IMS successfully closes the logs, you do not need to use the Log Recovery utility.
- In DUP mode, the utility reads an OLDS and duplicates all readable records on a log.
- In REP mode, the utility reads the interim log created in DUP mode, copies good log blocks, and replaces error blocks with good ones based on information that you specify in control statements. The output log data set is a usable OLDS.

If the OLDS being recovered has not been closed (DBRC shows a stop time of zero), you must use the output from REP mode as input to CLS mode.

SLDS Recovery

You must close an SLDS before you can use it as input to any utilities or to an IMS restart. The Log Recovery utility closes an SLDS created by an IMS batch job.

The Log Recovery utility recovers the SLDS from the following types of errors:

- An I/O error while reading the input log data set
- An error in the log record
- A log record sequence error

To recover an SLDS, the Log Recovery utility does the following:

- In DUP mode, the utility reads an SLDS and duplicates all readable log records onto an interim log. When the utility encounters the first error or end of file (EOF), it stops copying and closes the SLDS. When you specify a nonzero error count, DUP mode writes error blocks and error ID records on the interim log until either the error count reaches the number you specify or EOF is reached. The utility uses the interim log containing these error blocks and error ID records as input to REP mode.

  You can close an SLDS by using DUP mode and specifying the error count as zero. REP mode is not required.
- In REP mode, the utility reads the interim log created by DUP mode, copies good log blocks, and replaces error blocks with good ones based on information that you specify in control statements. The output log data set is a usable SLDS.

CQS Log Recovery

For OS/390 Release 2 and earlier, you have a limit of 168 offload data sets for the MVS system log. You cannot recover if the last offload data set is full and CQS fails. Be sure to schedule CQS system checkpoints frequently enough to prevent your offload data sets from filling.

For OS/390 Release 3 and later, you can define groups of offload data sets, and OS/390 uses each group in turn so that you do not run out of data sets.

WADS or RDS Log Recovery

You cannot recover a WADS or RDS. If a problem occurs with either type of data set, you must scratch, reallocate, and format it during an IMS restart.

Dependent Region Failures

Two types of dependent region failures are:

- Application program failures, in which the application program abends but the region is still active.
- Region controller failures, in which the region fails. This kind of failure can occur, for example, when the program loops.

If an LU 6.2 conversation established by the application program is running in the failing dependent region, APPC/MVS terminates the conversation with the DEALLOCATE abend.

Application Program Failures

When an IMS application program abends, IMS issues message `DFS554I` and, in some cases, message `DFS555I`.

IMS backs out all database changes made by the failing program to the last checkpoint, or to last point at which the program was scheduled. IMS stops the program and transaction, and you must use the /START PROGRAM and /START TRANSACTION commands to restart them after correcting the cause of the failure.

If an error occurs during database backout and restartable backout does not correct the problem, follow the procedures for database recovery before reactivating the program and transaction.

Region Controller Failures

One symptom of a dependent region failure is an increase in the number of transactions on the message queue. Monitor the status of your regions with the following commands: /DISPLAY ACTIVE, /DISPLAY Q TRANSACTION, and /DISPLAY Q BALGRP.

If the region appears to be looping, you should stop it with the /STOP REGION ABDUMP command. If the region does not stop, enter the /STOP REGION CANCEL command.

MVS issues messages IEF450I and IEF404I to indicate an abend in a BMP, MPP, or IFP region. If these messages are not followed by a DFS554I message, enter the /STOP REGION ABDUMP command so IMS recognizes that the region has terminated and backs out any database updates.

After you determine and correct the problem that caused the region to loop or that caused the abend, use the /START command to start the transaction and application program again.

Restarting BMP Regions

If the application program uses the IMS restart call (XRST), you can restart the program from its latest checkpoint. You can identify this checkpoint from one of the following messages:

DFS681I
This message lists the checkpoint ID and the PSB name, and is issued when the program takes a checkpoint.

DFS395A
The Database Batch Backout utility issues this message when it cannot restart.

DFS682I
This message lists the checkpoint ID and the program name, and is issued after emergency restart or after dynamic backout following an abend.

Specify the checkpoint ID value or LAST in the PARM field of the JCL used for restarting the BMP.

Restarting Fast Path Regions

If a Fast Path message-driven application program or DEDB online utility program abends, you can restart the Fast Path region immediately.

You should provide the MTO with documentation of the names of the procedures that are started using the /START REGION command because the procedure name must be given with this command. You should keep this documentation current because these regions require system resources, such as virtual storage and database buffers.

Database Failures

IMS issues message DFS0451I or DFS0451A when there is an error in an IMS database.

Every time IMS encounters a DL/I I/O error, IMS creates an extended error queue element (EEQE) that identifies the block or VSAM control interval in error.

When IMS closes a database, it automatically retries read and write errors on DL/I databases. If successful, forward recovery of the database is not required. Otherwise, forward recovery is eventually required. It might be possible to defer recovery to a more convenient time. Deferring recovery does not inhibit scheduling access or updating.

Using DEDB multiple area data sets also allows application programs to continue when I/O errors exist. For DEDB I/O errors, IMS issues messages DFS2571, DFS2572, DFS3712, and DFS3713. If a DEDB area is not available, the application receives an FH status code.

IMS maintains I/O information and buffer images across restarts. IMS does this by recording the EEQEs in DBRC, notifying all systems that are using the IRLM, and, during initialization and checkpoint, logging EEQEs and virtual buffers to the OLDS.

Database Write Errors

An IMS application program is unaware of a database write error. IMS does not pass a return code to the application program after a write operation completes or fails to complete. Instead, when a write error occurs, IMS creates an extended error queue element (EEQE), allocates a buffer for the block or control interval in error, and writes a log record with information about the buffer.

IMS uses the buffer for all subsequent I/O to the block or CI, including database updates and read requests. When IMS closes the database, it retries the original write operation that failed; if it is successful, IMS frees the buffer, removes the corresponding EEQE from DBRC, and notifies all subsystems to disregard their corresponding EEQEs.

There is a limit to the number of EEQEs that IMS can create. For Fast Path areas, an area is stopped after 100 EEQEs. For full-function databases, the limit is 32,767 EEQEs. Before encountering this limit, you would likely encounter virtual storage limitations for the buffers needed to write that many EEQEs.

You can defer recovery for weeks or longer, but deferring recovery too long can increase the time that IMS requires to perform the recovery.

IMS notifies DBRC of each EEQE; DBRC records each EEQE in the DBDS record of the RECON data set.

Database Read Errors

When a read error occurs, IMS returns an AO status code to the application program. IMS also creates an EEQE for the data in error but does not create a buffer for it.

Recovering a Database

If a database is physically lost or damaged in such a way that records in it become inaccessible, you can reconstruct the database from the information you have been keeping: image copies, logs, and so forth. This type of recovery is known as forward recovery.

Definition: *Forward recovery* involves reconstructing information and reapplying it to the database. It is based on the notion that if you knew what the data was like at one time and you know what you have done to it since then, you can process the data to return the database to the state it was in just before it was lost.

Figure 7 illustrates this concept. You know what the database was like at one time, because you made a backup copy of it. You know what changes (additions, deletions, alterations) have been made to the database since the backup copy was made, because IMS has been recording these changes on the log. Therefore, you only need to combine the two, and create a new data set to replace the database you have lost.

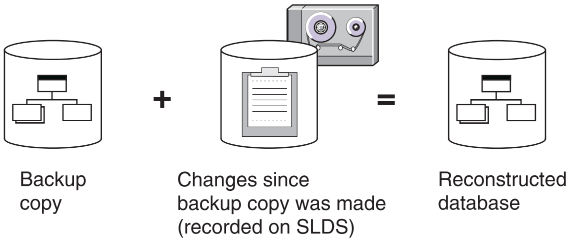


Figure 7. Database Reconstruction (Forward Recovery)

IMS supplies two methods for recovering databases:

- The database recovery service.
- The Database Recovery utility (DFSURDB0). Figure 8 on page 164 illustrates how the utility works. For more information about running DFSURDB0, see *IMS Version 9: Utilities Reference: Database and Transaction Manager*.

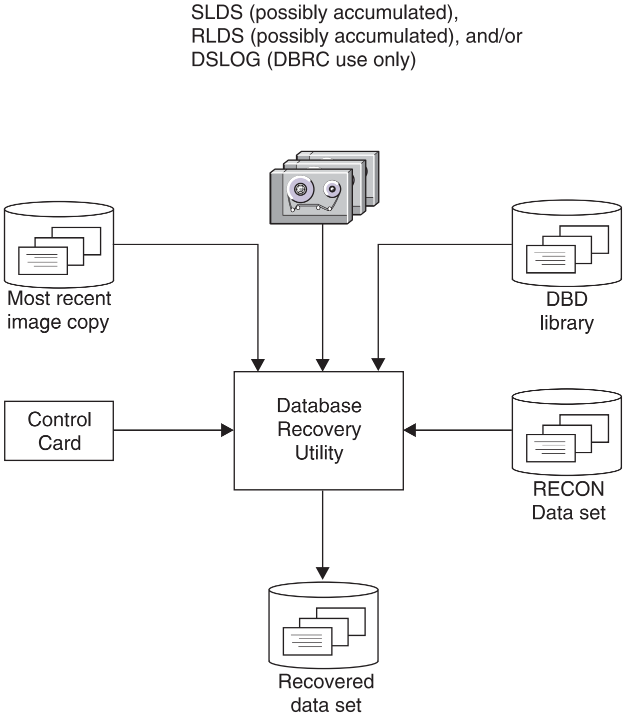


Figure 8. Using the IMS Utility DFSURDB0 for Database Recovery

Restriction: DBRC is unable to generate JCL for recovering databases that are not registered with DBRC. You can use the `LIST.LOG` command to produce a listing of current OLDSs, RLDSs, and SLDSs. Using this command lets you check manually that the JCL you are using for change accumulation or database recovery contains DD statements for all active log data sets. You can also use the `LIST.LOG` command when you use DBRC batch log control.

Recommendation: Do not supply logs or change accumulation input when running the Database Recovery utility for nonrecoverable DBDSs.

Level of Database Recovery

The Database Recovery utility and database recovery service perform database recovery at the data-set level. Usually, only a single data set of the database requires recovery. If more than one data set has been lost or damaged, you need to recover each one separately if you are using the Database Recovery utility. If you are using the database recovery service, all the lost or damaged data sets can be recovered in one recovery operation.

With DEDBs, using multiple area data sets reduces the need for recovery. If one copy has an I/O error, you can create another copy online and drop the error copy.

Concurrent Image Copy Recovery

A concurrent image copy is “fuzzy” because while IMS makes the image copy of the database, IMS might also make updates to the database:

- Before the start of the concurrent image copy, and IMS might not have written those updates to the DBDS yet
- While the concurrent image copy is being taken

After restoring the data set from the concurrent image copy, IMS applies changes which occurred while the image copy was being taken from the log. Therefore, to recover a DBDS that has fuzzy image copies, you might need to supply logs to the recovery job which have timestamps that precede the start of the start of the concurrent image copy. The DBRC `GENJCL.RECOV` command generates JCL with the correct logs listed.

If you use the database recovery service to recover a DBDS that has fuzzy image copies, the database recovery service automatically processes all the necessary logs required for the recovery.

For more information about concurrent image copies, see "Concurrent Image Copies" on page 62.

HSSP Image Copy Recovery

When you use an HSSP image copy for database recovery, you must tell the Database Recovery utility that you are supplying an HSSP image copy.

The HSSP image copy is usable as a fuzzy image copy whenever image copy completes successfully. If the image copy process fails during the creation of an image copy, IMS returns the image copy data set to the pool of available image copy data sets.

For more information about HSSP image copies, refer to "HSSP Image Copies" on page 63.

DL/I I/O Errors and Recovery

The way IMS handles DL/I I/O errors allows you to delay database recovery until it is convenient. If the errors do not recur or can be corrected automatically, recovery might even be unnecessary. IMS maintains data integrity by reconstructing information about I/O errors on later warm or emergency restarts.

When an I/O error occurs, IMS issues message `DFS0451I` and creates an extended error queue element (EEQE). When IMS successfully forward recovers a DBDS with an EEQE, DBRC updates the appropriate entries in the RECON data set, and IMS removes the EEQE.

You can use the `/DISPLAY DATABASE` command with the `BKERR` keyword to display information about the outstanding EEQEs associated with a database.

Related Reading: For more information about this command, see *IMS Version 9: Command Reference*. For more information about database I/O errors, see "Database Failures" on page 162. See also "DL/I I/O Errors during Backout" on page 72.

DEDB Full Condition

The database-full condition in a DEDB area indicates that one or more portions of the root-addressable area of the DEDB can no longer accept `ISRT` calls.

IMS detects the out-of-space condition for DEDB sequential dependent segments during sync-point processing. Although the application program may receive a warning status code, some `ISRT` calls, issued from a Fast Path message-driven program or from an MPP region, cause the region to abnormally terminate with a `U0844` abend code.

Recommendation: Track space usage within a DEDB area using one of the following methods:

- The application program can use information given by the position (POS) call
- The master terminal operator can monitor it using the appropriate /DISPLAY DB or QUERY DB command to show the total allocated and the total used CIs in both the root-addressable and the sequential-dependent portions of the area.

If a DEDB area does reach the database-full condition, Fast Path provides two online utilities that may negate the need for using an offline application program to perform a database unload and reload operation to resolve the problem. The two utilities are:

- The Sequential Dependent Delete utility, which deletes the sequential-dependent segments in all or part of the sequential-dependent portion of an area. This makes the space available for further sequential dependent inserts.
- The Direct Reorganization utility, which removes space fragmentation in an area. The utility runs against one DEDB area at a time.

  Recommendation: Although you can limit the reorganization to the one unit of work that did not allow inserts, you should reorganize more or all of the area. Any independent overflow space that is freed from one unit of work becomes available for use by any other unit of work within that area.

Continued Database Authorization

If IMS encounters a write I/O error on a DBDS, and the DBDS is registered with DBRC, IMS:

1. Builds an EEQE for the data that could not be written to the DBDS
2. Calls DBRC

If you register the database with DBRC, DBRC updates the RECON data set for the DBDS by:

- Flagging the DBDS record to indicate that recovery is needed
- Recording the address of the EEQE in the DBDS record
- Incrementing the DB record to count the EEQE

The recovery-needed flag prohibits new authorizations for some IMS utilities such as the Batch Image Copy utility. Also, if the recovery-needed flag is set and there are no EEQEs, DBRC denies all authorizations except for the recovery utilities.

If you do not register the database with DBRC, IMS assumes that the EEQEs that it builds from the log records during warm or emergency restart are correct. IMS discards EEQE information for nonregistered databases when you stop the database by using the /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command.

I/O Error Retry

When IMS closes a database after updating it, IMS automatically retries EEQEs for read and write errors. IMS authorizes attached subsystems to use a database with write errors, but IMS does not allow them to access blocks or CIs that were not successfully written. If any application program in one of these subsystems requests data requiring access to one of these blocks or CIs, IMS gives the program an AO status code, indicating a read error.

When the I/O-error retry is successful, IMS deletes the EEQE and sends messages to the MTO and MVS console. IMS sends message DFS0614I for each error that it successfully retries, and message DFS0615I when it resolves all outstanding errors.

Planning Your Database Recovery Strategy

You can recover databases in many different ways. The simplest, although possibly the most time-consuming, approach is to use an image copy plus all log data sets created since that image copy. You can also:

- Use the database recovery service
- Use the Change Accumulation utility to reduce the amount of log input to the Database Recovery utility
- Create an RLDS with the Log Archive utility and use the RLDS instead of an SLDS

The Database Recovery utility always processes log tapes **after** it processes the change accumulation data set. If, therefore, you must include log tapes that were created **before** the change accumulation data set, you must execute DFSURDB0 twice—once with the image copy and log tapes, and again with the change accumulation data set.

If you register your databases with DBRC, DBRC ensures the correct input is provided to the Database Recovery utility, and optionally generates the necessary JCL. For more detailed information on DBRC, see the *IMS Version 9: DBRC Guide and Reference*.

Some of the advantages and disadvantages of the different recovery techniques are described in the following sections.

Using Database Change Accumulation Input for Recovery

If you use the Database Change Accumulation utility (DFSUCUM0) periodically to consolidate your old log volumes, you can use these accumulated volumes as input to the database recovery service and the Database Recovery utility. You must include any log volumes recorded since the last time you ran the Database Change Accumulation utility as input to the recovery—either in their current format or after being passed through the Database Change Accumulation utility.

When you run the Database Change Accumulation utility, you can specify the DBDSs for which you want to accumulate change records. For each DBDS, you can either ignore change records or write them to the optional output log data set. By specifying a subset of the DBDSs (called a change accumulation group), you can reduce the size of the change accumulation data set, and improve performance of the Database Recovery utility. However, you also must run the Database Change Accumulation utility (and pass the log data set) for each change accumulation group.

If you select only one DBDS each time you run the Database Change Accumulation utility, you must run the utility (and pass the log data set) once for each DBDS. With this technique, each output change accumulation file contains only those records needed for the recovery of a single DBDS.

The following are some common recovery techniques:

- Define a single change accumulation group for all DBDSs.

  In this case, the Database Change Accumulation utility reads the log data set only once, but the Database Recovery utility must read records in the output change accumulation data set for potentially unnecessary databases. This technique improves the performance of the Database Change Accumulation utility, but degrades performance of the database recovery.
- Define several change accumulation groups.

In this case, you divide your databases into groups of several databases, often by volume or by application. If you take image copies frequently, you can keep the number of records scanned by the Database Recovery utility to a minimum.

- Define no change accumulation groups. Run the Database Change Accumulation utility, but not as a regular job.

  When you need to recover a database, specify that the utility accumulate changes for the affected DBDS only. Provide all log data sets created since the latest image copy of the data set to be recovered. If you use one of the image copy utilities, you must include the log created when the image copy was made as input to the Database Change Accumulation utility.

  If you are considering this approach, you should make some test runs to see whether the sum of the execution times of the Database Change Accumulation and Database Recovery utilities is less than the time needed to run the Database Recovery utility with unaccumulated log data sets.

Using Log Data Sets for Recovery

If you decide not to use the Database Change Accumulation utility, consider using an RLDS produced by the Log Archive utility. RLDSs are smaller than SLDSs because they contain only those records needed for database recovery. The frequency of image copies has a major impact on recovery time in this case, depending on the volume of database updates.

You must submit the log volumes to the recovery utility in chronological sequence: from the oldest to the most recent.

If you back out a database using the Database Batch Backout utility after making an image copy of it, you must use both the log created by the Batch Backout utility and the original log as input to the Database Recovery utility. This is because the Batch Backout utility backs out some of the changes recorded on the original log.

Using DBRC to Supervise Recovery

Recommendation: Use DBRC to supervise database recovery. By using DBRC, your task of recovery is greatly simplified.

During daily operations, DBRC keeps track of the activity of the database: backup copies taken, relevant log volumes recorded, change accumulations made, and so forth. Thus, DBRC knows what to supply to the recovery utility to recover the database: image copies and log volumes. DBRC generates the necessary JCL to run the utility and guarantee the proper input in the proper order.

Related Reading: For more information on DBRC, see *IMS Version 9: DBRC Guide and Reference*.

Recovering Databases with DBRC

DBRC supports two types of database recovery for DBDSs and area data sets (ADSs):

- *Full recovery* restores a DBDS or ADS to its current state as recorded in the RECON data set.
- *Time-stamp recovery* restores a DBDS or ADS to the contents it had at the time you specify.

Full Recovery: Full recovery of a DBDS or ADS is generally a two-step process. You must restore a copy of the DBDS or ADS, then apply changes made subsequent to that copy. These changes can be contained in change accumulation data sets and log data sets. If you make backup copies of your databases, running the Database Recovery utility can accomplish both steps.

Time-Stamp Recovery: Time-stamp recovery of a DBDS or ADS involves recovering a DBDS or ADS to some previous point in time, usually when the DBDS or ADS was not being updated. Generally, you perform time-stamp recoveries to recover from logic errors, such as bad input data, or an operational error, such as duplicate execution of a batch job. A time-stamp recovery has the effect of backing out one or more of the most recent sets of updates.

Recommendation: Perform time-stamp recoveries with great caution. When recovering one DBDS or ADS, you must perform similar recoveries for all related DBDSs or ADSs. Examples of such related data sets include not only those connected through logical relationships, but also indexes and databases containing multiple data set groups. Because DBRC does not know how DBDSs or ADSs are related, you must be sure to perform all related time-stamp recoveries.

Recovering DEDBs: The general strategy for recovering a DEDB is to use the Change Accumulation utility and then perform forward recovery. To assist in obtaining valid input, you can use the Log Recovery utility to create a copy of the OLDS that excludes the incomplete DEDB updates. You can also create a data set with only DEDB recovery records. Use the DEDB Area Data Set Compare utility to assist in the repair of a damaged area data set.

For more information about these utilities, including when to use them, see *IMS Version 9: Utilities Reference: Database and Transaction Manager*.

Recommendation: For an area that uses the shared VSO option, do not bring the area back online or restart failed IMSs that owned the connections until after you ensure that there are no failed-persistent XES connections to the CF structure used by the area. You can use the MVS SETXCF command to delete the connections.

Denial of Authorization for Recovery Utility

If DBRC denies authorization for an IMS recovery utility to run, you might receive message DFS3710A, which usually indicates that a previous recovery utility abended while processing and did not release its database authorization. To recover in this situation, delete the subsystem record in the RECON data set using the following command sequence:

1. `CHANGE.SUBSYS SSID(name) STARTRCV`
2. `CHANGE.SUBSYS SSID(name) ENDRECOV`
3. `DELETE.SUBSYS SSID(name)`

Nonrecoverable Databases

You can define any full-function DL/I database or Fast Path data entry database (DEDB) as nonrecoverable. You must identify nonrecoverable databases to DBRC by using the INIT.DB or CHANGE.DB commands with the NONRECOV keyword. A nonrecoverable DEDB must be registered with DBRC.

Defining a DL/I Database as Nonrecoverable

IMS treats a database as nonrecoverable if **all** of the following conditions are met:

- The database and its associated DBDSs are registered with DBRC.
- The database is defined as nonrecoverable.
- DBRC is running.

IMS does not log the “after” images of the database changes for nonrecoverable databases. However, IMS does still log “before” images. “Before” images are required for backouts, including:

- Dynamic backouts following abends and deadlocks
- Backouts performed during IMS restart
- Resync backouts performed during the abort of an in-doubt unit of work
- Deferred or restartable backouts performed because a /START DB or UPDATE DB START(ACCESS) command was issued when a backout process failed to back out a database

For any of the types of backouts, IMS backs out changes for nonrecoverable databases if the “before” images are available. This data is not available on archived log data sets because IMS does not archive the “before” images for nonrecoverable databases. IMS can back out changes for nonrecoverable databases in the following situations:

- Dynamic backout and resync backout back out changes for nonrecoverable databases because in-doubt data is logged during each checkpoint.
- Backout during IMS restart backs out changes for nonrecoverable databases if the log it is accessing is an OLDS. Otherwise, IMS does not back out the changes and issues messages.
- Deferred backouts back out changes for nonrecoverable databases if the data is in the checkpoint data or if the log they access is an OLDS.
- Batch backout backs out changes for nonrecoverable databases if the log supplied to it is not an archived log data set.

If a nonrecoverable database becomes corrupted, and a backout was performed, IMS issues a message to restore the database. If you suspect that the backout process failed to back out the database, use DBRC to restore the database. See *IMS Version 9: DBRC Guide and Reference*.

Defining a Fast Path DEDB as Nonrecoverable

In addition to defining a full-function database as nonrecoverable, you can define a Fast Path DEDB as nonrecoverable. This nonrecoverable option is available for both VSO and non-VSO DEDBs. A nonrecoverable DEDB is for use as a “work database” where recoverability is not a requirement. Because nonrecoverability is designated at the database level, all areas of the DEBD are nonrecoverable.

Change logging to the OLDS is suppressed for all areas of a nonrecoverable DEDB. Because the IMS log contains no DEDB change logging, REDO cannot be performed for emergency restart or XRF takeover. At emergency restart or XRF takeover, nonrecoverable DEDB areas that were opened and updated, but with writes to DASD that did not complete, are left in an incomplete state.

Nonrecoverable DEDB areas that have been left in an incomplete state after an /ERE or XRF takeover may have to be restored before they can be used. Put operating procedures in place to define what needs to be done at emergency restart and XRF takeover when nonrecoverable DEDBs are in use, especially in a sysplex data sharing environment.

Network Failures

If IMS stops responding to a remote terminal, you might be able to recover the session if the IMS master terminal is still active. Use the /DISPLAY command to list the status of the following:

- Use the /DISPLAY LINE command for communication lines
- Use the /DISPLAY LINE PTERM command for terminals
- Use the /DISPLAY NODE command for nodes

The MTO should receive a message that indicates why the terminals are not operational if they have not been stopped by one of the following commands: /CLSDST, /STOP, or /PSTOP. Correct the error and issue a /START or /RSTART command to restore operation for the terminals. For a VTAM terminal session, you might only need to use the /OPNDST NODE command to restart it.

If you have a hung node, you can free it with a /CLSDST FORCE command, if the following conditions exist:

- A VTAM display indicates that no session exists.
- The output from a /DISPLAY NODE command indicates that a CID exists, the node is connected, and the node is not idle.

  If a session does exist, terminate it with the VTAM command VARY INACT,FORCE, but use caution, because this command terminates all parallel sessions.

If all terminals are operational and appear to function, then it is likely that an application program has terminated abnormally, or is in a loop, or has a resource conflict that prevents its scheduling. To find the cause of the problem, use one of the following commands: /DISPLAY ACTIVE, /DISPLAY PROGRAM, /DISPLAY TRAN, or QUERY TRAN. See Table 66 on page 381 for the appropriate keywords for the QUERY command.

If you use /DISPLAY ACTIVE and continually see a program listed as active in a MPP or BMP region, that program is probably in a loop. For a program that is continually active, check that its PROCLIM value (specified in the PSB) is correct. Use the /STOP REGION ABDUMP command to terminate the region, or use the /STOP command to stop programs and transactions. You should be careful when using the /STOP REGION CANCEL command because in certain instances it can cause an abend of the IMS control region.

CPI Communications Failures

For a CPI Communications driven LU 6.2 conversation, IMS TM is not involved and places no restrictions on your choice of either committing or backing out updates.

Session Failure

In the event of a session failure, a CPI Communications driven application program can provide full integrity by issuing commit (SRRCMIT) or a backout (SRRBACK) calls.

If an LU 6.2 session fails during an LU 6.2 conversation, you can end the conversation or continue processing.

If the session fails before the transaction completes phase one of the two-phase commit sync-point protocol, IMS takes action as follows:

- For both standard and modified standard DL/I application programs, if the original conversation is asynchronous, IMS commits all database changes, sends all output messages asynchronously, and deallocates any established conversations.
- For standard DL/I application programs:
  - If the conversation is synchronous and a user destination exit routine is provided, the routine determines whether to abort or commit. If it commits, IMS sends all output messages asynchronously and commits all database changes. If it aborts, or if no user destination exit routine is provided, IMS aborts the transaction and backs out all changes.
- For modified standard DL/I application programs:
  - If the user destination exit routine commits, IMS commits database changes and inserts all messages to the IMS message queue asynchronously.
  - If the original conversation is synchronous and the user destination exit routine aborts, IMS aborts the transaction, backs out all database changes, and discards all messages inserted to the IMS message queue (unless the insert used an express PCB).

If the session fails after the transaction starts phase two of the two-phase commit, IMS sync-point processing continues for both standard and modified DL/I application programs, despite a session failure.

System Failure

When the IMS system fails during a CPI Communications driven LU 6.2 conversation, IMS determines whether to resolve in-doubts for IMS-protected resources. Examples of IMS-protected resources are IMS DB databases, DB2 databases, and IMS TM messages.

For standard and modified standard DL/I application programs, if the IMS system fails before the transaction completes phase one of the two-phase commit sync-point protocol, IMS backs out changed data during IMS restart. This backout includes all discardable transactions that were processing at the time of the system failure. IMS requeues non-discardable transactions so they can be processed.

If the transaction completes phase one of the two-phase commit, IMS resolves in-doubts during IMS restart. If only IMS resources are affected, IMS commits changes. If DB2 resources are affected, IMS tells DB2 to commit or abort the changed data, as appropriate.

Recovery Processing for CPI Communications Driven Application Programs

IMS provides no recovery processing for application programs that use the explicit CPI application programming interface (API). IMS discards all messages for and from CPI Communications driven application programs at IMS restart, regardless of their state at the time of failure. Application program designers should use the SAA® resource recovery resynchronization functions for explicit CPI application programs.

If your application programs require recovery assistance, they should use the implicit API provided by IMS rather than the CPI API. The implicit API support is the original IMS DL/I API (using *xxx*TDLI calls, where *xxx* represents the programming language used). Implicit support allows application programs that do not normally use LU 6.2 protocols to use LU 6.2 devices.

Related Reading: For more information about IMS DL/I calls, see *IMS Version 9: Application Programming: Database Manager*. For more information about CPI Communications, see *CPI Communications Reference* and *SAA CPI Resource Recovery Reference*.

MSC VTAM Message Resynchronization and Recovery

For IMS systems linked by VTAM, IMS maintains the integrity of messages and control blocks during system and session failures. IMS recovers sessions automatically if both half-sessions save resynchronization information.

Because a logical link is broken after an abend caused by a system failure, user action is required for recovery. Performing a normal cold start destroys session resynchronization information, so you should use the `/ERESTART` command to restart IMS.

During emergency restart, IMS restores the message queues. After the link is reestablished, IMS resynchronizes messages between half-sessions.

In contrast, the message queues are lost during a cold start because IMS cannot restart the link at the point at which the system failed. Messages that were lost must be resubmitted.

After you reestablish a VTAM link after a session fails, you must reassign as primary the half-session originally assigned as primary (the one from which you issue the `/RSTART` command).

An MVS system can run more than one IMS subsystem at a time. One IMS subsystem communicates with the other subsystem using real-storage-to-real-storage communication. MVS provides the real-storage-to-real-storage connection primarily for system protection and testing purposes.

If a physical link between systems fails, the MTO should reestablish communication through an alternate physical link:

1. Use the `/MSASSIGN LINK` command to reassign the logical link to an alternate physical link. This command must be entered on both systems.
2. Use the `/RSTART LINK` command to start the logical link.

CQS Failures and Structure Failures

The CQS subsystem automatically recovers from most types of failures. Because it is registered with the MVS Automatic Restart Manager, if the CQS subsystem fails, MVS restarts it. If one of the structures in the coupling facility fails, CQS initiates a structure rebuild (using data from the CQS log) to recover the structure.

If you lose connection with the last remaining coupling facility, IMS stops sharing the message queues and, because there are no queues available for input or output, soon thereafter stops work. After the connection is restored, CQS reconnects to the structures and rebuilds them if necessary, and IMS resumes work.

In the case of a resource structure, a coupling facility list structure, CQS does not support structure recovery, structure checkpointing, or overflow processing. To provide optimum performance, CQS does not log changes to resource structures. But to aid in recovering resources, you can define resource structures in the CFRM policy to duplex them automatically.

CCTL Failures

The most common types of CCTL failures are:

- CCTL region failures, in which a program is not stopped
- CCTL thread failures, in which a program is stopped

CCTL Region Failure

If a CCTL fails, it normally disconnects from the IMS DBCTL system and has no effect on the IMS DBCTL system itself. The CCTL threads are terminated immediately or after they complete their current DBCTL request. These thread terminations appear as DFS554I messages. The programs are not stopped.

What action IMS takes for database changes made (units of recovery), depend on the sync-point state of the thread. If a CCTL fails, a U113 abend of the DBCTL control region can occur.

Related Reading: For more information on what to do for CCTL failures, see *IMS Version 9: Customization Guide* or your CCTL documentation.

CCTL Thread Failure

A CCTL thread failure can occur if the CCTL thread makes a database resource adapter (DRA) request, and IMS abends during that request. When this happens, IMS issues a DFS554I message, and the CCTL indicates the failure to its operator. The program is stopped; you can start it again using the /START command after the problem is resolved. If a CCTL thread fails within the CCTL itself, IMS takes no action.

A thread can also fail because of an abend while the thread is within IMS or the DRA. For example, when the IMS DBCTL system fails, all CCTL threads are abended in the DRA with a U002 abend. What happens to the UOR associated with the CCTL thread is determined by its sync-point state. If the failure occurs while UOR is in-flight, IMS backs the UOR out; if the UOR is in-doubt (in the middle of the sync-point process), the UOR remains in-doubt. See “Two-Phase Commit in the Synchronization Process” on page 38.

CCTL Thread Looping

Use CCTL commands to determine if the CCTL is looping or if it has issued a DBCTL request and is waiting for a response. Refer to your CCTL documentation if the CCTL is looping. Otherwise, issue the /DISPLAY command to find the region with the recovery token that corresponds to the CCTL task that is waiting for a response from DBCTL. Then use the /STOP REGION command to force that thread to end.

DBCTL Failures

A termination of a IMS DBCTL system does not cause connected subsystems to terminate; the subsystems are simply left without DBCTL services. The state of the DRA is determined by how the CCTL responds to notification of the IMS DBCTL system failure.

If any of the DBCTL address spaces (DBC, DBRC, or DLISAS) fails, all of its address spaces are terminated.

Normally, you terminate the IMS DBCTL system using a `/CHECKPOINT FREEZE` command, but you can also use the MVS `MODIFY` command to force termination, particularly if an IMS shutdown or failure results in a loop. If you use the `MODIFY` command and want a dump of the address space, use the `DUMP` keyword; otherwise use the `STOP` keyword. The DBCTL control region terminates with a `U0020` abend, and IMS issues the following messages: `DFS628I` and `DFS629I`. These messages tell you that an abend is scheduled and what the job name is.

After determining the cause of the failure and correcting it, restart the IMS DBCTL system using the `/ERESTART` command. The CCTL cannot reconnect to the IMS DBCTL system until the `/ERE` command completes. The CCTL has several options on how to respond to a DBCTL failure; for details see *IMS Version 9: Customization Guide* and your CCTL documentation.

When a CCTL connects to an IMS DBCTL system after a restart, they resynchronize in-doubt UORs automatically. Resynchronization means that the CCTL requests an action (commit, abort, or forget) and the IMS DBCTL system processes that request. If the commit or abort process fails, IMS takes the same actions as in the DB/DC environment. If the commit process fails for systems with DEDBs, IMS sends message `DFS2282I` to the DBCTL operator. For all failures, IMS removes the EEQEs because the UOR is no longer in-doubt.

If a CCTL requests resynchronization of an in-doubt UOR, and the DBCTL subsystem has no knowledge of that UOR, IMS returns code 218 to the CCTL, and sends message `DFS2283I` to the DBCTL operator. The DBCTL operator can use the `/DISPLAY` command to display the in-doubt UORs known to the IMS DBCTL subsystem.

Related Reading: See "Recovery in an IMS DBCTL Environment" on page 7 and "Two-Phase Commit in the Synchronization Process" on page 38 for more information.

IRLM Failures

When an IRLM fails, IMS subsystems using the IRLM cannot continue normal operation. IMS terminates active programs using the IRLM with abend `U3303` and puts their messages on a suspend queue. For wait-for-input programs using the IRLM, IMS either terminates them with abend `U3303` on their next database call or gives them a `QC` status code. IMS inhibits subsequent program scheduling until IMS reconnects with the IRLM.

Before IMS can back out the IRLM failure, every dependent region with an intent to use a database must terminate its threads. IMS cannot terminate or reconnect to the IRLM until all dependent regions disconnect from the IRLM. Therefore, you must abnormally terminate any region that has not terminated (such as those programs that are waiting in an for a timer expiration or a for a reply to a Write-to-Operator with Reply (WTOR) message.

Restart the IRLM using the MVS `START` command. Then, reconnect IMS subsystems to the restarted IRLM using the MVS `MODIFY` *`jobname`* `RECONNECT` command. IMS automatically dequeues messages on the suspend queue.

Recovery with Data Sharing

In a data-sharing environment, you must extend IMS recovery procedures by protecting surviving subsystems from database records that contain incomplete changes left by a failing subsystem.

IMS recovers a shared database by restoring the database from the latest image copy. You can use the merge function provided with the IMS Database Change Accumulation utility (DFSUCUM0) to merge the log data sets from the subsystems that updated the database. Apply the records from the utility (in the output data sets) for a forward recovery. If only one subsystem in a data-sharing environment updated the database, IMS does not need to use the sequence numbers and the Database Change Accumulation utility, because it has only one set of log data sets to manage.

Related Reading: For more information about the Database Change Accumulation utility, see *IMS Version 9: Utilities Reference: Database and Transaction Manager*.

IMS data sharing also introduces other challenges. Two IMS Internal Resource Lock Managers (IRLMs) might lose communication with each other because:

- An XCF might fail.
- An IRLM might fail, leaving its IMS subsystems running.
- An MVS system might fail, bringing down the IRLM, VTAM, DBRC, and IMS subsystems running under it.

In a data-sharing environment, IMS must ensure data integrity after failure of one or more components of data sharing.

The IRLM and DBRC work together with IMS to protect shared databases before and after a failure leaves incomplete changes in these databases. This protection remains after a failure until failed subsystems complete backout of incomplete changes.

DBRC and Protecting Data

DBRC provides database-level (level 1) data sharing and database-level protection after a failure occurs. DBRC prevents conflicts in database allocation that could compromise database integrity.

DBRC uses the RECON data set to record information about each database authorization it grants to IMS subsystems, and to indicate whether a subsystem is updating a database. DBRC removes this indication when any of the following occur:

- The subsystem terminates normally.
- Dynamic backout completes after a failure.

If an IMS subsystem fails during an update, the RECON data set still indicates this incomplete update. DBRC does not grant additional database authorizations unless the IRLM retains locks that protect the incomplete changes.

The IRLM and Protecting Data

If the IRLM is present and remains running, it retains locks on incomplete changes at the database record level after an IMS subsystem fails. The IRLM removes these locks when dynamic backout completes.

If the IRLM fails while its IMS subsystems continue to run, the subsystems stop all work in progress, dynamically back out all incomplete database changes, and call DBRC to relinquish their authorizations to shared databases. After you restart the IRLM, the subsystems resume processing.

If two or more IRLMs are active during block-level data sharing, and one IRLM fails with or without the MVS system that it runs under, the surviving IRLMs and their IMS subsystems reauthorize with DBRC the use of all shared databases. In this case, because the surviving IRLMs know about the block-level locks (that protect the incomplete changes made by the failed subsystem), they permit their IMS subsystems to continue block-level sharing of the affected databases.

Fast Database Recovery

In a sysplex data-sharing environment, if one IMS subsystem fails while it holds locks on data in shared databases, the other sharing IMS subsystems must wait for the failed IMS to restart and release its locks. To reduce the amount of time the sharing subsystems must wait, you can use IMS Fast Database Recovery (FDBR) regions.

An FDBR region monitors an IMS subsystem and can automatically recover database resources (shared databases and areas) if that IMS subsystem fails.

Restriction: You cannot use an FDBR region with the following types of subsystems: IMS DCCTL, XRF, or DBCTL standby.

An FDBR region tracks a single IMS subsystem, so you must set up a separate FDBR region for each IMS subsystem in the sysplex that you want FDBR to track. An FDBR region does not have to run in the same MVS system as the IMS subsystem it tracks, but both the FDBR region and the IMS subsystem must be in the same XCF group. The FDBR region must also have access to the following IMS data sets:

- ACBLIBx
- MODBLKSx
- MODSTATx
- OLDS
- RDS
- RECONx
- SDFSRESLx
- WADS

IMS Fast Database Recovery resembles XRF in that it operates in distinct phases:

- Surveillance
- Recovery process
- Post-recovery

FDBR Surveillance

During the surveillance phase, the FDBR region monitors database activity by reading the IMS log. It monitors all activity for shared databases and areas, and can monitor activity for nonshared areas. An FDBR region also monitors RSR-related log activity at an active site.

Restrictions: An FDBR region does not track MSDB activity. It also does not track databases at an RSR tracking site.

If the IMS subsystem that the FDBR region is tracking fails, the FDBR region automatically begins recovery for all tracked databases and areas. If the IRLM for the tracked IMS subsystem fails, or if there is no log activity from the tracked IMS subsystem for a specified length of time, the FDBR region issues a message telling the operator to begin FDBR recovery.

FDBR Recovery Process

If the tracked IMS subsystem fails, the FDBR region automatically begins recovery. To begin recovery manually, issue the MVS `MODIFY` *fdbrproc*`,RECOVER` command.

During automatic recovery, the FDBR region uses the MVS Availability Manager to determine when I/O is complete for the databases and areas it will recover. For manual recovery, the FDBR does not wait for I/O to complete because it assumes the I/O is already complete.

During recovery, the FDBR region does the following for all tracked databases and areas:

- Allocates and opens database data sets
- Backs out updates for full-function databases
- Performs REDO processing for DEDB areas

After recovery is complete, the FDBR region tells IRLM to release retained locks for the recovered databases and areas. The FDBR region then issues a message that recovery is complete.

If the tracked IMS subsystem’s XCF fails, the FDBR region stops tracking and issues a message. If the FDBR region’s XCF fails, the region abnormally terminates.

FDBR Post-Recovery

When recovery is complete, all shared databases and areas are available to other sharing IMS subsystems in the sysplex. You can restart the failed IMS subsystem at any time.

During IMS restart, IMS does the following to complete recovery:

- Writes updates for shared VSO areas to DASD
- Recovers nonshared DEDB areas that were not tracked by the FDBR region
- Recovers MSDBs

After the IMS subsystem is running, you can reestablish tracking by the FDBR region by issuing the MVS `START` *fdbrproc* command.

Deciding Where to Implement FDBR Regions

As an IMS system programmer you must decide where to implement FDBR regions on your systems. Consider these two requirements:

- There can be one FDBR region for each IMS subsystem in the data sharing group.
- The FDBR region should be on a separate MVS system from the IMS that it tracks. This is to protect the FDBR region during an MVS system failure on the system that contains the tracked IMS.

You can configure FDBR in several ways. Figure 9 on page 179 shows FDBR regions placed with IMS subsystems. Each FDBR region tracks an IMS subsystem on a different MVS system. This configuration has the advantage of low resource requirements.

In this configuration, if IMS1 fails, FDBR1 detects the failure and recovers any databases that IMS1 was using when it failed. If MVS1 fails (with both IMS1 and FDBR4), FDBR1 recovers the database resources for IMS1, and FDBR4 can be established on another system until MVS1 is restarted.

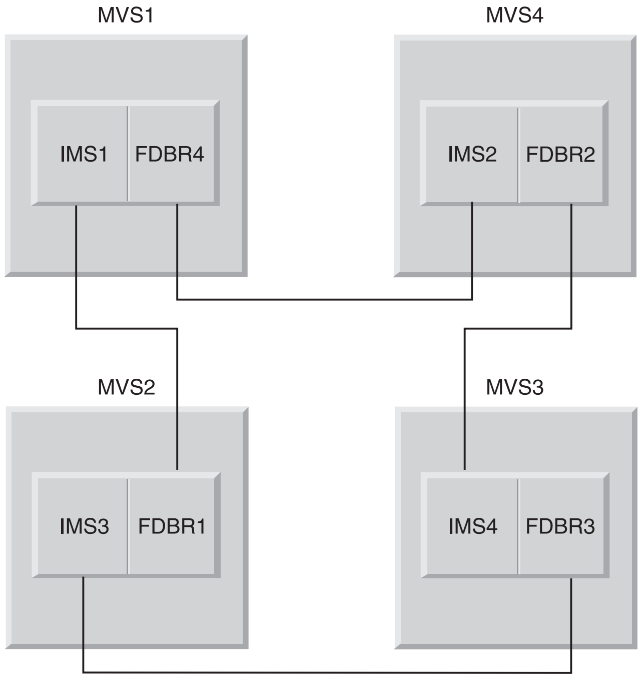


Figure 9. System Configuration 1 for FDBR Tracking of an MVS Subsystem

Figure 10 on page 180 shows all FDBR regions on a separate MVS system from the tracked IMS subsystems. This configuration requires an additional hardware resource (the extra MVS system). If an MVS system failure occurs on MVS1, MVS2, or MVS3, none of the FDBR regions are affected.

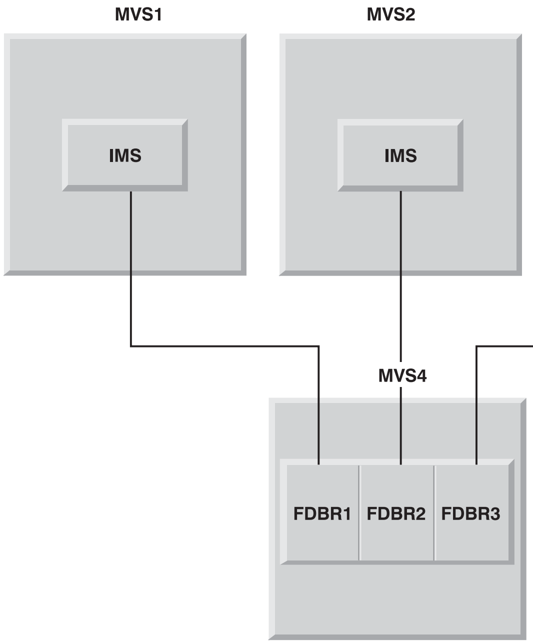


Figure 10. System Configuration 2 for FDBR Tracking of an MVS Subsystem

Defining and Tailoring an FDBR Region

The FDBR procedure initiates an FDBR region. This procedure is similar to those used to define IMS subsystems (for example the IMS procedure, or the DBC procedure).

Enabling an IMS Subsystem for FDBR

DB/DC: To enable a DB/DC subsystem for FDBR, specify the FDRMBR parameter in the IMS procedure. The FDRMBR parameter defines the DB/DC system as FDBR-capable.

Restriction: If you specify both FDRMBR and HSBID (XRF configuration) parameters in the DBC procedure, the FDRMBR parameter is ignored.

You must supply the IMS ID of the system that FDBR will track. You must specify this IMS ID in the control statement for the DFSFDR*xx* IMS.PROCLIB member, and it must match the ID specified for the IMSID EXEC parameter.

DBCTL: To enable a DBCTL subsystem for FDBR, specify the FDRMBR parameter in the DBC procedure. The FDRMBR parameter defines the DBCTL system as FDBR-capable. Also specify the IMS ID of the system that FDBR will track in the control statement for the DFSFDR*xx* IMS.PROCLIB member, and it must match the ID specified for the IMSID EXEC parameter.

Restriction: If you specify both FDRMBR and DBRSE (DBCTL standby configuration) parameters in the DBC procedure, the FDRMBR parameter is ignored.

Creating the IMS.PROCLIB Member (DFSFDR*xx*)

The DFSFDR*xx* member specifies the options used by FDBR. IMS.PROCLIB can contain multiple instances of DFSFDR*xx*, but each DFSFDR*xx* member in IMS.PROCLIB must have a unique, two-digit suffix. FDBR, IMS, or DBC procedures identify which DFSFDR*xx* member to use.

Related Reading: For the details of these procedures, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

Using FDBR

This section describes the commands for using FDBR. For each command, `fdrproc` represents the FDBR procedure name.

Starting Tracking: Normally, you start FDBR with a JCL job, submitted during system start, after the tracked IMS is started. You can also issue the MVS START command to start FDBR tracking of an IMS subsystem, as follows:

```
S fdrproc
```

Obtaining Status: Use the MVS `MODIFY` command to obtain FDBR status, as follows:

```
F FDR1,STATUS
```

This command returns the following information:

```
DFS000I:  PHASE:   TRACKING  LOG-TIME:  17:48:41  FDR1
DFS000I:  ACT-ID:  SYS3      GROUPNAME: FDRSYS3   FDR1
DFS000I:  TIMEOUT: 060 SEC   AREA01:    NORECOV   FDR1
DFS000I:  SVSOOPEN: SERIAL   FPBUFF:    LOCAL     FDR1
```

The returned information includes:

PHASE
This field indicates the phase of the FDBR region. You can display the following phases:

INIT (initialization phase)
TRACKING (tracking phase)
RECOVERY (recovery phase)

ACT-ID
The IMS ID that FDBR is tracking. If the FDBR region is in initialization phase, N/A is shown.

GROUPNAME
XCF group name used for XCF monitoring. If the FDBR region is in initialization phase, N/A is shown.

TIMEOUT
XCF timeout value. If the FDBR region is in initialization phase, N/A is shown.

LOG-TIME
Time associated with the log record currently being processed by FDBR region. The length of time the FDBR lags behind the IMS in reading the log is the difference between the current time, as shown by the time stamp and the log time. If the FDBR region is in initialization phase, N/A is shown.

AREA01=
This field shows the share level 0|1 DEDB area recovery option as follows:

NORECOV
NORECOV option is specified in DFSFDR*xx* IMS.PROCLIB member.

RECOV
RECOV option is specified in DFSFDR*xx* IMS.PROCLIB member.

Initiating a Recovery: Use the MVS `MODIFY` command to initiate FDBR. The following command starts FDBR immediately:

```
F fdrproc,RECOVER
```

You can use this command if FDBR determines a time-out status during XCF or log surveillance. The command is not accepted if FDBR is being initialized or is already in a recovery process.

Terminating Tracking: Use the MVS `MODIFY` command to terminate FDBR. The following command stops FDBR tracking activity:

```
F fdrproc,TERM
```

The command is not accepted if FDBR is being initialized or is in a recovery process.

Stopping the FDBR Region: You can use two commands to stop the FDBR region functions. They are both MVS `MODIFY` commands.

1. The following command ends FDBR functions without producing a dump:

   ```
   F fdrproc,STOP
   ```

   FDBR ends with a return code of 0200.
2. The following command ends FDBR functions and produces a dump:

   ```
   F fdrproc,DUMP
   ```

   FDBR ends with return code of 0020.

User Access Problems

The task of determining and responding to user access problems can be done by the IMS MTO, the network support group, or a user liaison group.

If your installation has a user liaison group that helps end users with problem determination, you need to explain what information that group should obtain from the end user before they contact the IMS MTO. This information includes a description of the problem's symptoms (such as poor response time or no response at all), details of any system or application program error messages received, terminal status lights, and so forth. Be sure to give the user liaison group guidelines on whether to call the network support group, application support group, or IMS MTO.

After they have gathered all information from the end user, they should develop a plan for determining the cause of the problem and, if possible, correcting it. Typically, user problems can be classified as:

- Poor system response
- No response, or terminal keyboard locked
- Unexpected output (system or application program error messages)

If the general system response is poor, the MTO should follow the procedures described in "Monitoring the System" on page 95.

If the end user is getting no response at all or has a locked keyboard, the problem may lie in the network or in the status of the application program resources required by the transaction. For example, an OLDS may need to be archived to free additional logging resources. Use one of the following commands to determine if any of the required resources are unavailable:

- `/DISPLAY STATUS`
- `/DISPLAY OLDS`
- `QUERY DB STATUS`

If the end user is receiving application program error messages or invalid output, the MTO should follow your procedures for contacting the appropriate application support group. You should define the documentation that MTO need to provide to the support groups, including a dump of the message region (if one was produced), or exact details of the original transaction the user entered. For IMS or VTAM system messages, the MTO should follow the appropriate error recovery procedures.

Chapter 13. Automated Operations

IMS provides the following tools to help you automate your operations:

- Time-Controlled Operations (TCO), available for IMS DB/DC and DCCTL environments
- Automated Operator Interface (AOI), of which there are two types:
  - Type 1, available for the IMS DB/DC and DCCTL environments
  - Type 2, available for all IMS environments (DB/DC, DCCTL, and DBCTL)
- REXX SPOC (Single Point of Control) API (application programming interface), available in an IMSplex environment

IMS operations can also be automated using NetView. Because NetView is independent of IMS, it can gather information and issue commands that are not available to IMS. For more information about NetView, see *Tivoli NetView for z/OS Installation: Getting Started*.

See “Automated Operations” on page 18 for an introduction to automated operations for IMS. This chapter describes the primary tools for automating operations, TCO and AOI, in some detail, but more details about TCO and AOI are in other books in the IMS library; look for the “Related Reading” labels.

What Is TCO?

TCO allows you to initiate time-driven procedures for any IMS operation. This relieves the operator of any task that is performed routinely or at a specific time each day. TCO can issue any IMS command that an operator can issue manually, except the `/NRESTART` and `/ERESTART` commands. TCO can start and stop telecommunication lines and message regions, monitor periodic system status, and notify users of system status. TCO uses scripts that call specific exit routines that schedule transactions, execute commands, execute message switches, and deliver deferred output.

TCO is upwardly compatible with and replaces IBM’s Time Initiated Input Facility (TIIF) Field Developed Program.

What TCO Can Do

TCO can generate any IMS input that an IMS operator can. Specifically, TCO can:

- Execute time-initiated commands, transactions, and message switches.

  Operators usually execute transactions and message switches manually at specific times. TCO can perform these functions and can also execute commands at specified times. Using TCO eliminates errors caused by mistyping commands.

  Restriction: TCO cannot execute the IMS restart commands, `/NRESTART` and `/ERESTART`, or initiate conversational transactions. TCO also does not support full-function response mode or Fast Path input transactions.
- Start automatically during IMS initialization and terminate when IMS shuts down.

In addition, you can:

- Start, stop, and change TCO scripts without disrupting operations.

  After starting IMS with the time-initiated function, you can:
  - Stop TCO processing

  - Add, delete, change, or replace a TCO script
  - Start processing an existing script
  - Begin processing a new script
  - Start TCO if it was not started automatically at IMS restart

  These actions do not affect IMS operations.
- Restart the initial script DFSTCF during IMS restart.

  When IMS is restarted, TCO loads the initial script, called DFSTCF, for processing. If you stop TCO before IMS terminates, TCO will not start processing when IMS restarts.
- Test TCO scripts without interfering with online execution or shutting down IMS.

  Using the TCO Verification utility ensures IMS processes only error-free scripts and can do so without disrupting normal operations. See “Using the TCO Verification Utility” on page 193 for more information.

The following are possible implementations of TCO facilities:
- Starting IMS resources
- Handling peak loads
- Scheduling low-priority and non-IMS jobs
- Handling operations across multiple time zones
- Monitoring the system
- Updating user status
- Scheduling different procedures for different days
- Starting large IMS TM networks in phases to prevent resource contention or deadlocks
- Shutting down IMS

TCO Components

TCO consists of:
- An initialization routine that performs initial housekeeping and starts the TCO timer service
- A TCO timer service that processes startup requests and timer requests, calls an exit routine, and waits for future requests
- A device-dependent module that passes the input generated by the timer service to the IMS control region
- An offline script-member verification utility that ensures that IMS only processes error-free scripts
- A termination routine that performs cleanup and tells the timer service when to terminate

You also need an exit routine. This can be the exit routine IMS provides or an exit routine you write.

TCO and the exit routines execute as a subtask of the IMS control region. An error in the exit routine terminates TCO only, and does not disrupt IMS online operation.

Installing TCO

There are no IMS system definition parameters for TCO. Everything you need for TCO is automatically included when you install IMS.

In order to initialize TCO when IMS starts, include a DFSTCF DD statement in the IMS procedure. You can use the data set name IMS.TCFSLIB, or you can use another data set for your TCO scripts.

IMS defines two logical terminals for TCO: DFSTCF and DFSTCFI. These terminal definitions allow TCO input to appear as though it came from a terminal. You do not need to specify either LTERM name in your IMS stage 1 input. If you do, IMS issues a G979 message telling you that your stage 1 includes duplicate LTERM names.

You can use the IMS Security Maintenance utility (SMU) to control which IMS commands the TCO LTERMs can issue. List the secured commands for LTERM DFSTCFI in the SMU input.

If the IMS startup parameter RCF=A or RCF=S is specified at system startup, the SMU is not used for authorizing commands from static terminals (including the TCO input LTERM DFSTCFI). Instead, you can call RACF by specifying TCORACF=Y and call the Command Authorization Exit Routine (DFSCCMD0), if it exists. To sign on a user ID to IMS, you can add a /SIGN ON command at the beginning of the TCO script and a /SIGN OFF command at the end of the script. RACF checks all subsequent commands to determine if the signed-on user is authorized to issue the commands. The commands are then passed to the DFSCCMD0 exit routine.

Using TCO

TCO operates by processing a script that is an unblocked 80-character member in a partitioned data set. The script contains time schedule requests and, optionally, messages that are passed to the TCO exit routine.

At the time specified in a time schedule request, IMS loads a TCO exit routine and schedules it to perform a task. If the TCO exit routine generates input, IMS passes this input to the TCO device dependent module, which then passes the data to IMS using standard DL/I interfaces. If the data is a command, IMS executes it. If it is a transaction or a message switch, IMS enqueues it for processing.

IMS displays output generated from the input, including commands, on any non-ETO device you specify. Thus, you can log all input and output for automatically-generated commands or other activity.

IMS loads and processes TCO scripts and exit routines in the IMS control region. So, you can modify either the script or the TCO exit routine without stopping and restarting IMS or disrupting operations.

The following sections describe possible uses of TCO.

Starting IMS Resources Automatically

After IMS starts, TCO can execute any non-restart commands. It can start message regions and telecommunication lines, and automatically display the system status.

Handling Peak Loads

The TCO exit routine can assign extra message regions or terminate message regions that you no longer need based on the time of day or the status of the IMS message queues.

Scheduling Non-IMS Jobs

TCO can execute non-IMS-related jobs at a specific time by issuing the /START REGION *procname* command. For example, you can specify that:

- At 8:00 p.m., TCO notifies users that an application will terminate.
- At 8:15 p.m., TCO stops transactions for the application.
- At 8:16 p.m., TCO starts the non-IMS job.

Scheduling Low-Priority Jobs

You might have some types of transactions that are processed only once or twice a day, usually by BMPs. By collecting these transactions in the message queues at 0 priority, TCO can schedule and process the transactions at less-busy times. For example, assuming your operators leave for lunch at 12:00 p.m., you can specify that:

- At 12:05 p.m., TCO issues an `/ASSIGN` command to reassign the transactions' priority (greater than 0), so processing can begin. If a the transactions need a BMP, TCO can generate one using the `/START REGION` command.
- At 12:50 p.m., TCO reassigns the transactions' priority to 0.

By the time the operators return from lunch, processing can continue for high-priority jobs.

Handling Multiple Time Zones

If some of your IMS systems are in one time zone, for example US Eastern-standard time, and other systems are in another time zone, for example US Pacific time, the TCO exit routine can automatically start and stop terminals and other resources, taking the time difference into account.

Monitoring the System

TCO can automatically maintain periodic displays of the pools, message queues, and system status without operator intervention.

Updating User Status

You can automatically update the status or location of user terminals on a regular basis.

Starting Large Networks in Phases

TCO can start a large teleprocessing network in phases to prevent IMS, VTAM, or NCP resource overloads. By issuing a command for a few nodes on each physical line rather than all nodes on a few physical lines, the network starts more quickly because of increased parallel activity.

Example: While IMS is starting, TCO can issue an `/OPNDST` command to start 100 nodes. A minute later, TCO can issue another `/OPNDST` command to start another 100 nodes. Two minutes after startup, TCO can start another 100 nodes, and so on, until the entire network is started.

Shutting Down IMS

The TCO exit routine can notify users that IMS will shut down, and, for example, 10 minutes later, it can stop the telecommunication lines and message regions. TCO can even issue the `/CHECKPOINT` command to shut IMS down.

TCO Exit Routine

The TCO exit routine inserts messages on the IMS message queue for processing. The messages are the commands, transactions, and message switches that you specify in the time schedule requests and message sets that make up a TCO script. The TCO exit routine passes any data found in columns 56 through 71 of the time schedule request to IMS for processing.

You do not need to write your own exit routine. You can schedule predefined commands, transactions, and message switches at predefined times using DFSTXIT0, the TCO exit routine IMS supplies.

You must bind user-written exit routines with the TCO language interface module (DFSTDLI0) and place them in the IMS.SDFSRESL data set. A user-written exit routine can have any name.

Example: The following example shows a user-written exit routine, MYEXIT, being bound with DFSTDLI0:

```
//XIT       JOB ...
//BIND      EXEC PGM=IEWL
//SYSUT1    DD UNIT=SYSDA,SPACE=(TRK,(20,20))
//SYSPRINT  DD SYSOUT=A
//SYSLMOD   DD DSN=IMS.SDFSRESL,DISP=SHR
//INLIB     DD DSN=IMS.OBJ,DISP=SHR
//SYSLIN    DD *
    INCLUDE  INLIB(MYEXIT)
    INCLUDE  INLIB(DFSTDLI0)
    NAME     MYEXIT(R)
/*
```

In this example, IMS.SDFSRESL is an authorized library that contains all load modules. IMS.OBJ is a library that contains all object modules. The job finds the object modules, MYEXIT and DFSTDLI0, in IMS.OBJ and places the result of the bind in IMS.SDFSRESL.

To load and execute a user-written exit routine, a time schedule request in a TCO script that is executing must refer to the routine. The following example shows a time schedule request in a TCO script that executes the user-written exit routine, MYEXIT:

```
*TIME 1200 MYEXIT
```

- Columns 1-5 indicate that this is a time schedule request.
- Columns 7-10 indicate that the request starts at 12:00 p.m.
- Columns 12-19 indicate the name of the user exit routine, in this case MYEXIT.

TCO CNT Edit Exit Routine

You can use the TCO CNT Edit exit routine DFSTCNT0 to restrict users (LTERMs) from loading TCO scripts. When IMS loads a TCO script, IMS calls the CNT exit routine to validate the load request. If the routine rejects the load request, IMS sends an error message to the terminal attempting to load the script.

You do not need to write your own exit routine if you do not want to restrict any users from loading TCO scripts. IMS provides a default exit routine, DFSTCNT0, that is always called unless it is replaced with a customized exit routine. The default exit routine allows all users to load TCO scripts.

You can write your own exit routine to authorizes certain users (LTERMs) to load TCO scripts. You must name the exit routine DFSTCNT0 and replace the default routine.

| IMS uses the DFSTCNT0 exit routine instead of the Message Switching (Input) Edit exit routine when the message switching destination is DFSTCF.

| IMS does not use SMU security or RACF to determine authorization loading for TCO scripts.

Time Schedule Requests

A time schedule request is a statement in a TCO script that tells IMS to process a particular task at a specified time. A time schedule request also specifies which exit routine is scheduled and what data should be passed to the exit routine.

A time schedule request can specify:

- A time of day, by hour and minute.
- A range of times, for example, every 10 minutes from 10:00 to 16:00.
- IMS startup. Requests are scheduled immediately after IMS restart, or when a new script is loaded.
- A specified delay after IMS startup, for example, 10 minutes after startup.

A time schedule request either contains 16 bytes of data or refers to a variable-length message set that contains one or more single segment or multi-segment messages.

Example: Figure 11 shows two message sets and time schedule requests for broadcasting a message. The example also shows a scale to indicate the alignment of the fields.

```
.........1.........2.........3.........4.........5.........6.........7.........8

/BRO LTERM CTRL                                                        S00000001
THE IMS TIME-CONTROLLED OPERATION IS UP.                                00000002
*TIME      DFSTXIT0            S                       ****             00000003
/BRO LTERM CTRL                                                        S00000004
FIVE MINUTES HAVE ELAPSED SINCE STARTUP, AND TCO IS ACTIVE.             00000005
*TIME      DFSTXIT0      0005  S                       ****             00000006
```

Figure 11. Example of Message Sets and Time Schedule Requests

In this example, the first message set is:

```
/BRO LTERM CTRL
THE IMS TIME-CONTROLLED OPERATION IS UP
```

The presence of the S in column 72 indicates the end of a segment and that the next segment is part of the same message. If there are any errors in this message set, the numbers will be added to the error message. The time schedule request indicates that this broadcast should take place as soon as the script is initialized.

The second message set in this example is:

```
/BRO LTERM CTRL
FIVE MINUTES HAVE ELAPSED SINCE STARTUP, AND TCO IS ACTIVE
```

The associated time schedule request indicates that this message should be broadcast 5 minutes after the script is initialized.

Required Fields for Time Schedule Requests

The required fields for a time schedule request specify the following: the time you want the task to be performed, and to which exit routine to pass the script.

- Columns 1-5 (ID) IDENTIFICATION

  Must contain the characters *TIME. All other requests are treated as message sets.
- Columns 12-19 (NAME) USER EXIT ROUTINE NAME

Must include the one- to eight-character name of the exit routine in the IMS.SDFSRESL data set. If the name has fewer than eight characters, it must be left-justified and padded to the right with blanks. NAME must begin in column 12.

Optional Fields for Time Schedule Requests

By specifying one or more optional fields, you can request tasks to be processed during IMS startup or at a specified time following startup. You can also specify for processing to start, stop for an interval, and then restart.

- Columns 7-10 (HHMM) START TIME

  The time the task should start. A start time is required unless the request is for startup. Valid values for HH are 00 to 23, and valid values for MM are 00 to 59.
- Columns 21-24 (HHMM) END TIME

  The time the task should end. Use this field only if you specify an interval time in columns 26-29. Otherwise, the field is ignored. This time must be a later time than the start time specified in columns 7-10. Valid values for HH are 00 to 23, and valid values for MM are 00 to 59.

  If you request multiple dispatching for a task, the end time also stops the multiple dispatching. If you do not specify an end time, multiple dispatching continues until the end of the day (2359).
- Columns 26-29 (HHMM) INTERVAL TIME

  This field specifies the amount of time you want to defer a request. The interval time can be any time after the start time (or startup) to a time before this request is to be dispatched. Valid values for HH are 00 to 23, and valid values for MM are 00 to 59.
- Column 31 (Flag) RESIDENT INDICATOR

  Specifies whether the exit routine is resident in memory. If you specify any request for an exit routine as resident, all requests use the resident copy. The options for this column are:

  Blank The exit routine is resident and paged in and out as required.

  D The exit routine is loaded each time a request is generated.

  There is an advantage and a disadvantage to specifying the exit routine as resident:

  - Advantage: the request is performed more quickly because the system does not have to load the routine each time it is needed to process a time schedule request.
  - Disadvantage: the resident copy is always used; IMS does not load a new copy. If you want to load a new copy of the exit routine, you must either change the resident indicator on the time schedule request statements to “D”, or reload the TCO script using `DFSTCF LOAD`.
- Column 32 (Flag) DISPATCH INDICATOR

  Specifies when the request is dispatched (started). The options for this column are:

  Blank The request is dispatched each day at the start time specified in columns 7-10. The start time in columns 7-10 is required.

  O The request is dispatched the first day the script member is loaded. It is not dispatched again until the TCO script is reinitialized. A start time in columns 7-10 is required.

  S The request is dispatched as soon as the script member is initialized, either when IMS starts or when the script member is loaded. The start time, if specified in columns 7-10, is ignored.

If you specify an interval time in columns 26-29, the request is delayed from startup before the request is scheduled. With the “S” option, the request is processed only once each time IMS is initialized or the script-member is reloaded.

- Columns 33-35

  Reserved
- Columns 56-71 MESSAGE FLAG

  This field includes 16 bytes of data to be passed to the exit routine, plus the time of day. You can choose any data, or you can leave the field blank. If you set this flag to ****, the previous message set, if any, is passed to the exit routine instead of the 16 bytes found in the time schedule request.
- Columns 73-80 LINE NUMBER

  Sequence numbers for the requests. If IMS or the TCO Verification utility detects an error, the number found in columns 73-80 is included in the error message to help you to identify the cause of the error.

  Recommendation: Include sequence numbers for all commands and message sets.

Message Sets

A message set is made up of one or more messages in a TCO script-member to be passed to an exit routine for processing. A message can consist of one or more message requests.

Definition: A message request is an 80-character string that defines data to be passed to the exit routine specified in a time schedule request.

Required Fields for Message Sets

- Columns 1-71 MESSAGE DATA

  IMS commands to be executed. Columns 1-5 must not contain “*TIME” (unless you want to specify a timed schedule request).
- Column 72 CONTINUATION

  Allows you to include single-segment or multi-segment messages and allows segments or messages to be longer than 71 characters.

  The options for this column are:

  Blank Indicates the end of the segment and the end of the message. The exit routine assumes that the next statement begins a new message. At the end of the message, you must include a time schedule request with “****” in columns 56-59.

  S Indicates the end of the segment only (that is, that the next statement begins a new segment of the same message).

  X Indicates the segment is continued in the next statement. Multiple statements are combined before the segment is passed to the exit routine. Only nine statements can be continued, for a total of 10 statements and 710 characters. You must be careful not to exceed the large message queue buffer size.

Optional Field for Message Sets

- Column 73-80 LINE NUMBER

Sequence numbers for the requests. If IMS or the TCO Verification utility detects an error, the number found in columns 73-80 is included in the error message to help you to identify the cause of the error.

Recommendation: Include sequence numbers for all commands and message sets.

When IMS reaches the end of the TCO script, or at the end of the day, IMS starts again at the beginning of the script. Time schedule requests for startup are not reexecuted, unless you reload the script. Requests with the dispatch indicator set to O (onetime execution) are not rescheduled unless the script is reloaded.

Statements do not need to be in time sequence in the script, nor do startup requests need to be first in the file. If TCO finds multiple startup schedule requests, or more than one time schedule request for the same time, TCO processes them in the order they are listed.

Using the TCO Verification Utility

Use the TCO Verification utility, DFSTVER0, to ensure your TCO scripts are error free. You should run the utility before you execute any script online. The utility detects any script errors that would occur during online execution, except an error caused by insufficient storage.

You must add a TCO script to the script library before you can execute the TCO Verification utility. You can verify more than one script at a time by assigning an input control statement for each script you want to verify.

Related Reading: For more information about executing DFSTVER0, see *IMS Version 9: Utilities Reference: Database and Transaction Manager*.

Loading a TCO Script

You can process a new TCO script without stopping (and restarting) IMS or TCO by sending a message switch to the TCO LTERM, DFSTCF. An operator, a program, or an exit routine can issue the command to load a new script. When you loaded a new script, and IMS processes it, IMS disregards all previous schedule requests.

Being able to load TCO scripts dynamically allows you to do the following:

- Change or correct TCO scripts without stopping IMS
- Chain multiple scripts, so that they are processed on different days

The format of the load command is:

```
►►──DFSTCF──LOAD──membername──┬─────────────────────────────┬──────────────────►◄
                              └─OUTPUT──lterm──CONT──count─┘
```

LOAD *membername*
: Specifies the load request and the one- to eight-character script member name.

OUTPUT *lterm*
: Specifies that you want IMS to send the output from the command to a specified LTERM, rather than the default LTERM, the IMS MTO. *lterm* specifies the one- to eight-character LTERM name to which IMS sends output.

CONT *count*
Specifies how many segments (continuation lines) a message can have. The default is 9. Use this keyword to change the maximum size of a message. Valid values for *count* can be 1 to 99.

You must include a blank space between each keyword and parameter.

Stopping TCO

To stop TCO, enter one of the following commands:

```
/PSTOP LTERM DFSTCF
/PSTOP LTERM ALL
```

To restart TCO, use one of the following commands:

```
/START LTERM DFSTCF
/START LTERM ALL
```

Related Reading: For more information about these commands, see *IMS Version 9: Command Reference*.

What Is AOI?

The IMS Automated Operator Interface (AOI) is made up of several IMS functions you can use to manage IMS processing more efficiently. AOI lets an application program, using DL/I calls, issue most IMS operator commands and receive command responses. AOI lets you write an exit routine to intercept:

- IMS system messages routed to the MTO
- Commands
- Command responses

Using AOI, you can examine these messages, commands, and command responses and issue a message to any terminal or to the IMS message queue to be processed by an application program.

You can use AOI to develop commands tailored to your installation and to automate some portions of the recovery process. AOI can relieve the MTO of numerous controlling and monitoring duties. This is especially useful if you manage a large terminal network.

Using AOI:

- Relieves operators from repetitive command entry.
- Increases accuracy for command entry, especially when the commands have LTERM and telecommunication line control parameters.
- Helps you control IMS resources at prearranged times, such as during region startup and database image copy.
- Lets you prompt the MTO to take specific actions.
- Allows entry of commands with long operand lists. This can be important for commands that are larger than can be entered from a terminal.

Automated operator programs can:

- Supervise terminals
- Control IMS monitor and trace operations
- Help detect database problems

- Assist in recovery by issuing the commands `/STOP DB`, `/LOCK DB`, and either the `/DBRECOVERY DB` or `UPDATE DB STOP(ACCESS)`

AOI Functions

The IMS AOI consists of an exit routine and application program DL/I calls. The exit routine is called the *Automated Operator (AO) exit routine*. An application program that issues the DL/I calls is called an *AO application program*. The term AOI refers to the overall function, rather than to either of its two parts.

IMS calls the AO exit routine continually for system messages, commands, and command responses. The AO exit routine can:

- Examine IMS messages, commands, and command responses
- Manipulate IMS messages
- Send messages to an AO application program

The AO application program lets you:

- Retrieve messages sent from an AO exit routine
- Issue various IMS operator commands
- Retrieve responses to those commands

You can use the AO exit routine and application program together or independently of each other. For example:

- You could use just an AO exit routine to monitor IMS messages and delete unwanted messages from the master terminal (MVS system console in the DBCTL environment).
- You could use just an AO application program to issue a set of IMS commands, perhaps to start programs or databases. When used alone, you start the AO application program just like any other IMS application program.
- You could use both the AO exit routine and AO application program together. The exit routine could, for example, direct a message to the application program. The application program could then do specific processing based on the message it receives from the exit routine. In this case, the AO exit routine starts the application program.

AOI can be used in any IMS environment (DB/DC, DBCTL, or DCCTL). But, depending on the environment, there are some differences in what AOI can do and what you must do to use it. See “AOI and the IMS Environments” on page 197 for a description of using AOI in different IMS environments.

Uses of AOI

You can use AOI for many different tasks, for example:

- Discarding unwanted messages
- Setting up a batch window
- Restarting application programs after an abend
- Customizing messages
- Creating an audit trail
- Supervising terminals

Discarding Unwanted Messages

You can write an AO exit routine to prevent the master terminal (MVS system console in the DBCTL environment) from being flooded with unwanted IMS messages. You can still log discarded messages to the secondary master terminal (by issuing the /SMCOPY command from an AO application program) in case, for example, the messages are subsequently needed for problem determination.

Setting Up a Batch Window

You can write an AO application to set aside time (usually each night) for running batch jobs, that is, to set up a batch window. When you run batch jobs, you usually take the databases offline. To set up your batch window using an AO application program, you can write a multiple step job that:

- Issues a /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command to take the database offline (BMP step)
- Issues a /DISPLAY DB or QUERY DB command to determine when the database is actually offline (BMP step)
- Initiates your batch work and does routine operational tasks, such as making image copies (batch step)
- Issues a /START DB or UPDATE DB START(ACCESS) SET(ACCTYPE()) command to bring the databases online (BMP step)

Restarting Application Programs after an Abend

When an application program abends, IMS issues a message and immediately stops the program. You can write an AO exit routine that sends the message to an AO application program that can then issue the /START PROGRAM command to allow work to continue immediately, rather than waiting for the MTO to restart the program.

Customizing Messages

You can write an exit routine that customizes IMS system messages before they are routed to the master terminal. You can delete parts of messages that have no meaning for the MTO. You can reformat a message to conform to a format established at your installation. Or, you can add text to a message to tell an MTO what action to take in response to the message.

Creating an Audit Trail

You can use AOI to track system or user actions. For example, if you find that certain commands are not being used appropriately at your installation, you can intercept them with an AO exit routine. The AO exit routine can send a copy of the command to an AO application, which can in turn send the command, with pertinent user information, to the MTO, or copy it to the log for later analysis.

Supervising Terminals

If your network is extensive, you might make one or more terminal operators responsible for terminal availability. These operators also watch for communication problems and can reassign LTERMs to alternate devices. You can write an AO application program that performs the same tasks, and in effect, functions as a supervisory terminal. Either the MTO or someone entering the appropriate transaction from a remote terminal can schedule the AO application program.

Processing Messages, Commands, and Command Responses

Because AOI can manipulate system messages, commands, and command responses, this section defines these terms and explains how AOI processes them.

Processing System Messages

A *system message* is any IMS message that is not a direct response to a command. When IMS issues a system message, it sends a copy of the message to one or more of the following:

- IMS master terminal (MVS system console in the DBCTL environment)

- MVS system console
- IMS secondary master terminal

IMS sends all system messages destined for the IMS master terminal to the AO exit routine. IMS does not send copies of system messages destined for the MVS system console (in a non-DBCTL environment) or the secondary master terminal to the AO exit routine. The AO exit routine can change or delete a system message sent to the master terminal. In the DBCTL environment, the original copy of a changed or deleted message is lost.

Copies of a system message destined for the MVS system console or secondary master terminal are sent before they are sent to the master terminal.

Processing Commands

A *command* is any IMS operator command. IMS sends a copy of all IMS operator commands entered from a terminal the AO exit routine, but does not send the following:

- `/FORMAT`
- `/LOOPTEST`
- `/MSVERIFY`
- `/RELEASE`

In the IMS DB/DC and DCCTL environments, IMS also does not send the following commands:

- `/NRESTART`
- `/ERESTART`

In the IMS DBCTL environment, IMS also sends the two restart commands listed above to the AO exit routine.

IMS also sends copies of the following types of commands to the AO exit routine DFSAOE00:

- Commands issued by `ICMD` calls in an AO application program
- IMS internally-generated commands

IMS does not execute the command until after IMS sends it to the AO exit routine.

Processing Command Responses

After a command is executed, IMS generates a *command response*. If an AO exit routine expressed interest in the command, IMS sends a copy of the command response to the routine. Any asynchronous system messages IMS produces as a result of a command are not command responses; they are system messages.

IMS does not send copies of command responses for the following types of commands to the AO exit routine DFSAOE00:

- Commands issued by `ICMD` calls in an AO application program
- IMS internally-generated commands

To process a command response, the AO exit routine must check for message segments after the first one. The `/DISPLAY`, `/RDISPLAY`, and `/RMxxxx` commands all generate multi-segment response messages.

AOI and the IMS Environments

You can use one of two AOI functions, depending on which IMS environment you are operating in and what you want to do. The AOI function for the IMS DB/DC and

DCCTL environments is called *type-1 AOI*. The AOI function for the DB/DC, DCCTL, and DBCTL environments is called *type-2 AOI*.

Type-1 AOI uses the following DL/I calls:

- CMD to issue operator commands
- GCMD to retrieve responses to those commands
- GU and GN to retrieve messages routed to the AO application program from the AO exit routine

The AO exit routine you write in this environment is named DFSAOUE0. Type-1 AOI uses the IMS message queues for communication between an AO application and an AO exit routine.

Type-2 AOI uses the following DL/I calls:

- ICMD to issue operator commands
- RCMD to retrieve responses to those commands
- GMSG to retrieve messages routed to the AO application program from the AO exit routine

The AO exit routine you write in this environment is named DFSAOE00. Type-2 AOI does not use the IMS message queues for communication between an AO application and an AO exit routine; instead, it uses an AOI token (an eight-byte name that associates messages with AO application programs).

Although this chapter pairs the AO exit routine and application program together based on the environments in which they operate, you could use a single AO application program to issue calls and retrieve messages from both AO exit routines, DFSAOUE0 and DFSAOE00. You can write such an application program to run in the IMS DB/DC or DCCTL environment, but not in the DBCTL environment.

Using Type-1 AOI

Type-1 AOI consists of the AO exit routine, DFSAOUE0, and AO application programs that use the GU, GN, CMD and GCMD calls. Type-1 AOI is applicable for the IMS DB/DC and DCCTL environments.

Type-1 AO Exit Routine (DFSAOUE0)

IMS calls the type-1 exit routine, DFSAOUE0, for:

- System messages destined for the master terminal
- A subset of operator-entered commands
- Associated command responses

DFSAOUE0 can examine these messages and commands before IMS sends or executes them. IMS does not call DFSAOUE0 for system messages destined for terminals other than the master terminal or for certain commands and command responses.

DFSAOUE0 can handle both single-segment and multisegment messages. You can write DFSAOUE0 to do the following:

- Modify IMS system messages. In addition to modifying a message, DFSAOUE0 can add up to 20 bytes of text to the end of a message. A copy of the modified message can be sent to an alternate destination.

Messages issued by IMS after a command has been entered (for example, the DFS058“command in progress” message) are not really messages; they are command responses and cannot be modified.

- Delete an IMS system message and, optionally, send a copy of the message to an alternate destination.
- Ignore selected segments of a message or an entire message.
- Send a copy of system messages, commands, or command responses to an alternate destination.
- Send a new message to an alternate destination in response to receiving a system message, command, or command response.
- Modify a copy of a command or command response and send the copy to an alternate destination. Note that, unlike system messages, the exit routine can only modify a **copy** of commands and command responses.
- Enter a transaction to be placed on a message queue. This function lets DFSAOUE0 schedule an AO application program. For example, after receiving startup messages DFS680 and DFS994I, DFSAOUE0 could invoke an AO application program to enter a series of commands to start the network.

  Restriction: In a shared-queues environment, any transactions to be processed by the local IMS subsystem (rather than using the shared queues) must be defined as SERIAL. These transactions cannot be processed by other IMS subsystems in the sysplex.
- Request the edited command buffer (when the input is a command) to determine which command parameters succeeded or failed.

DFSAOUE0 can modify or delete system messages only. It can modify or delete the system message that the original destination (the master terminal) receives. It can also send a modified copy to an alternate destination. You cannot, however, use DFSAOUE0 to modify or delete the original command or command response. You can use it to modify a copy of a command or command response that it sends to an alternate destination, but not the copy that the primary destination receives.

Type-1 AO Application Program (GU, GN, CMD and GCMD)

An AO application program can:

- Issue DL/I calls to retrieve messages from DFSAOUE0 (or DFSAOE00)
- Issue a subset of IMS commands
- Retrieve the responses to those commands

Retrieving Messages (GU and GN Call): An AO application program receives control when IMS enqueues a message with the appropriate transaction code. The AO application program retrieves the message from DFSAOUE0 by issuing GU and GN calls.

Issuing Commands and Retrieving Command Responses (CMD and GCMD): The AO application program issues a command using the CMD call. You can only use the CMD call if you also use the I/O PCB. PCB status codes indicate the results of a CMD call.

If there are more than one command response segments, the first one is returned to the I/O area. Use the GCMD call to retrieve subsequent segments. This call, similar to a GN call, places subsequent command response segments in the I/O area.

Security: You can secure the CMD call using the IMS Security Maintenance utility (SMU) security, RACF (or an equivalent product), the IMS Command Authorization exit routine (DFSCCMD0), or a combination of RACF and the exit routine. With

SMU or with the AOI= parameter on the TRANSACT system definition macro, you can specify which programs can issue operator commands. With SMU, RACF, or DFSCCMD0, you can define which commands an AOI application program can issue.

Related Reading: For more information about AOI security, see *IMS Version 9: Administration Guide: System*.

Recovery: IMS puts messages sent from DFSAOUE0 to the AO application program on the IMS message queue. Therefore, all messages are recoverable at IMS restart.

Supported Application Program Environments: Table 29 shows the types of application programs that can issue `CMD` and `GCMD` calls, by IMS environment.

Table 29. CMD and GCMD Call Support by Application Region Type

	IMS Environment		
Application Region Type	**DB/DC**	**DBCTL**	**DCCTL**
DRA thread	No	No	N/A
BMP (nonmessage-driven)	No	No	No
BMP (message-driven)	Yes	N/A	Yes
MPP	Yes	N/A	Yes
IFP	No	N/A	No

Using Type-2 AOI

Type-2 AOI consists of the AO exit routine, DFSAOE00, and AO application programs that use the `GMSG`, `ICMD`, and `RCMD` calls. Type-2 AOI is applicable for all IMS environments (DB/DC, DCCTL, and DBCTL).

Type-2 AO Exit Routine (DFSAOE00)

IMS calls the type-2 exit routine, DFSAOE00, for:

- IMS system messages destined for the master terminal
- Commands entered from a terminal and the responses to those commands
- Commands issued from an AO application program using the `ICMD` call
- IMS internal commands

IMS does not call DFSAOE00 for:

- Command responses to internal commands
- Command responses to an AO application program `ICMD` call
- Commands issued from an AO application using the `CMD` call
- System messages for which the destination is not the master terminal

You can write DFSAOE00 to do the following:

- Modify the text of IMS system messages. DFSAOE00 can also add up to 20 bytes of additional text to the end of a message.
- Delete IMS system messages. In a DBCTL environment, if a message is deleted or modified, IMS does not keep the original message sent to the MVS system console.
- Direct any system message, command, or command response to an AO application program.

- Start a BMP job (for example, an AO application program). DFSAOE00 can issue SVC 34 to start a BMP. You can use the APARM parameter of the EXEC statement to pass information to the AO application program. The AO application program can issue the DL/I INQY call to retrieve the value specified for the APARM parameter. You can use the START command to override the APARM parameter of the EXEC statement.

DFSAOE00 must use IMS AOI callable services to communicate with an AO application program. AOI callable services let you:

- Direct messages, commands, and command responses to an AO application program
- Cancel messages previously directed to an AO application program but not yet sent

Related Reading: For details on callable services, see *IMS Version 9: Customization Guide*.

Restriction: You cannot use DFSAOE00 to modify or delete commands and command responses, including commands from the MVS system console or an application program, or internally generated commands.

Type-2 AO Application Program (GMSG, ICMD, and RCMD)

An AO application program can issue DL/I calls to:

- Retrieve messages from AO exit routine DFSAOE00 (or DFSAOUE0)
- Issue a subset of IMS commands
- Retrieve the responses to those commands

Retrieving Messages (GMSG Call): DFSAOE00 does not use the IMS message queues to route messages to an AO application program. Thus, your AO application program can retrieve messages when you cannot or do not want to use the message queues.

The AO application program retrieves messages using the GMSG call, which associates an AOI token name with a message. The application program passes an 8-byte AOI token to IMS, and IMS returns to the program only those messages associated with the AOI token. By using different AOI tokens, DFSAOE00 can direct messages to different AO applications programs. This use of the AOI token is similar to the use of transaction names using the other AO exit routine, DFSAOUE0.

The GMSG call, specifying an 8-byte AOI token, retrieves the first segment of a message associated with the token. Subsequent GMSG calls, with no AOI token specified, retrieve the second through the last segments of a multisegment message, one segment at a time. If a program needs all segments of a multisegment message, it must issue GMSG (with no AOI token) until all segments are retrieved before issuing another GMSG call with an AOI token. When the program issues a new GMSG call with an AOI token, IMS discards all remaining message segments from the previous call.

An application program can specify a wait in the GMSG call. If there are no messages currently available for the specified AOI token, the application is put into a wait state until a message is available. The decision to wait is specified by the program, unlike a WFI (wait-for-input) transaction where the wait is specified in the transaction definition.

Restriction: In a shared-queues environment, any transactions enqueued to an AOI token are processed by the local IMS subsystem (rather than using the shared queues) and cannot be processed by other IMS subsystems in the sysplex.

Issuing Commands and Retrieving Command Responses (ICMD and RCMD Calls): The ICMD call issues an IMS command and retrieves the first command response segment if one exists. The RCMD call retrieves subsequent command response segments. When an ICMD call results in a multisegment command response, you might not want to retrieve all segments. If you need all segments, you must issue RCMD calls until all segments are retrieved before issuing another ICMD call. When the program issues a new ICMD call, IMS discards all remaining message segments from the previous call.

Using AIBTDLI: The GMSG, ICMD, and RCMD calls do not require the AO application program to pass a PCB to IMS. The calls use the application interface block (AIB) interface (AIBTDLI), which lets an application issue DL/I calls without a PCB address. The AIBTDLI interface also lets IMS pass unique return and reason codes to the application in the AIB.

Security: You can secure the ICMD call using RACF (or an equivalent product), the IMS Command Authorization exit routine (DFSCCMD0), or both. RACF lets you specify which commands an application program can issue. DFSCCMD0 and RACF also allow you to check authorizations during ICMD call processing.

Related Reading: For more information about AOI security, see *IMS Version 9: Administration Guide: System*.

Recovery: The messages directed from DFSAOE00 to the type-2 AO application program are not written to the IMS message queues. Therefore, messages directed to the AO application program cannot be recovered during IMS restart.

Supported Application Program Environments: Table 30 shows the types of application programs that can issue GMSG, ICMD, and RCMD calls for different IMS environments. In addition, these calls are supported for CPI Communications driven applications programs.

Table 30. GMSG, ICMD, and RCMD Call Support by Application Region Type

	IMS Environment		
Application Region Type	**DB/DC**	**DBCTL**	**DCCTL**
DRA thread	Yes	Yes	N/A
BMP (nonmessage-driven)	Yes	Yes	Yes
BMP (message-driven)	Yes	N/A	Yes
MPP	Yes	N/A	Yes
IFP	Yes	N/A	Yes

Specifying the AO Exit Routine

Your AO exit routine must be a stand-alone, 31-bit, reentrant module.

You specify your exit requirements by linking the exit routines you want to use in the IMS.SDFSRESL concatenation. You can specify both DFSAOUE0 and DFSAOE00.

Summary of Exit Routine Processing

Figure 12 shows the processing that occurs for an IMS system message; Figure 13 on page 204 shows the processing that occurs for a command entered at a terminal.

In an IMS DB/DC or DCCTL environment, you can use both DFSAOUE0 and DFSAOE00. If both exit routines are loaded, IMS calls DFSAOE00 first. DFSAOE00 can process the message or command, or it can pass control to DFSAOUE0 to do the processing.

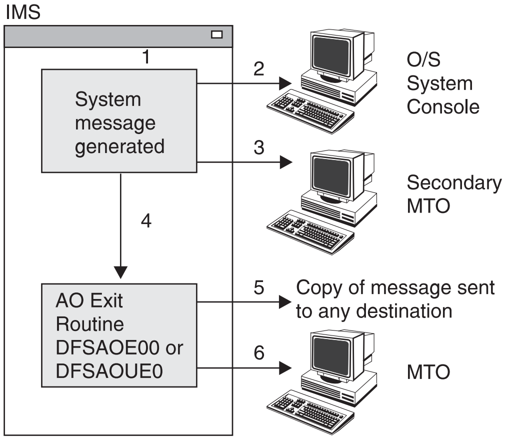

IMS
1
System message generated
2
O/S System Console
3
Secondary MTO
4
AO Exit Routine DFSAOE00 or DFSAOUE0
5
Copy of message sent to any destination
6
MTO

Figure 12. Processing for a System Message Using Both AO Exit Routines

The steps shown in Figure 12 include the following processing:

1. IMS generates a system message destined for the master terminal. Both AO exit routines have been specified.
2. Depending on the specific message, IMS sends a copy of the message to the MVS system console.
3. If you specify that a copy of the message be sent to the secondary master terminal, IMS sends a copy of the message (assuming the secondary master terminal exists).
4. IMS passes the copy of the message destined for the master terminal to DFSAOE00. DFSAOE00 determines which exit routine should process the message. When both exit routines are specified, IMS only calls DFSAOUE0 if DFSAOE00 determines that DFSAOUE0 should process the message.
5. The exit routine that processes the message can send a copy of it to any destination. If the exit routine is DFSAOE00, the destination is the AOI token, to which the message is enqueued. If the exit routine is DFSAOUE0, the destination is an LTERM or a transaction. The exit routine can alter or delete any segment of the message.
6. Unless the message has been deleted, IMS sends it to the master terminal.

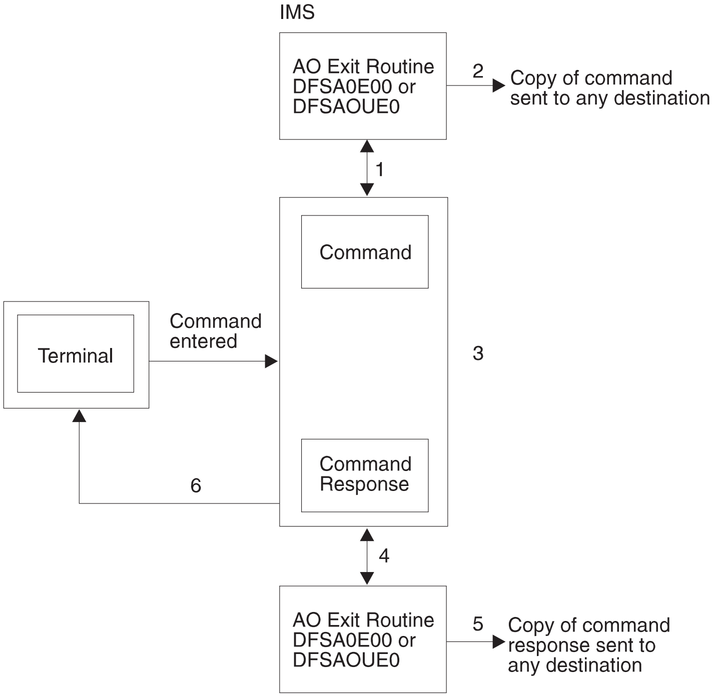

IMS
AO Exit Routine DFSA0E00 or DFSAOUE0
2
Copy of command sent to any destination
1
Command
Terminal
Command entered
3
Command Response
6
4
AO Exit Routine DFSA0E00 or DFSAOUE0
5
Copy of command response sent to any destination

Figure 13. Processing for a Command Using Both AO Exit Routines

The steps shown in Figure 13 include the following processing:

1. When a command is entered from a terminal, IMS sends a copy of the command to DFSAOE00 before executing the command. DFSAOE00 determines whether to process the command itself or have DFSAOUE0 process the command.
2. The exit routine that processes the command can send a copy of the command to any destination. If the exit routine is DFSAOE00, the destination is the AOI token, to which the message is enqueued. If the exit routine is DFSAOUE0, the destination is an LTERM or a transaction.
3. IMS executes the command and generates a command response.
4. If the exit routine requested a copy of the command response, before sending the command response, IMS passes it to whichever exit routine processed the command.
5. The exit routine that handles the command response can send a copy of it to any destination. If the exit routine is DFSAOE00, the destination is the AOI token, to which the message is enqueued. If the exit routine is DFSAOUE0, the destination is an LTERM or a transaction.
6. IMS sends the command response to the terminal that originated the command.

Comparison of Type-1 AOI and Type-2 AOI

- You can use type-1 AOI in the IMS DB/DC and DCCTL environments. You can use type-2 AOI in all IMS environments, DB/DC, DBCTL, and DCCTL.
- Type-1 AOI uses the IMS message queues; type-2 AOI does not.
- DFSAOE00 can process internal commands and commands from a type-2 AO application; DFSAOUE0 cannot.
- DFSAOUE0 can send a message to an alternate destination and request the edited command buffer when the message is a command; DFSAOE00 cannot.
- DFSAOE00 uses IMS AOI callable services; DFSAOUE0 does not.

- The ICMD call uses RACF (or equivalent), the Command Authorization exit routine (DFSCCMD0), or both, for security; the CMD call RACF (or equivalent), the Command Authorization exit routine (DFSCCMD0), or both, or the IMS Security Maintenance utility for security.
- An AO application program issues GU and GN calls to retrieve a message from DFSAOUE0; it issues GMSG calls to retrieve messages from DFSAOE00.
- MPP and message-driven BMP regions can issue CMD and GCMD calls in the IMS DB/DC and DCCTL environments. MPP, BMP, and IFP regions, DRA threads, and CPI Communications driven applications can issue ICMD and RCMD calls in the IMS DB/DC, DBCTL, and DCCTL environments.

For More Information on AOI

This chapter is intended as an introduction of the IMS AOI. The following books in the IMS library contain detailed information on AOI:

- *IMS Version 9: Customization Guide* describes both types of AOI, as well as AOI callable services.
- *IMS Version 9: Application Programming: Database Manager* and *IMS Version 9: Application Programming: Transaction Manager* describe the DL/I calls (ICMD, RCMD, GMSG, CMD, and GCMD) AO application programs can issue. These books also describe the status, return, and reason codes an application program can receive after a call.
- *IMS Version 9: Administration Guide: System* describes security for AO application programs.
- *IMS Version 9: Command Reference* lists the commands that an AO application program can issue.

Other reference information about AOI is in *IMS Version 9: Installation Volume 2: System Definition and Tailoring* and *IMS Version 9: Messages and Codes, Volume 1*.

What is SPOC?

IMS provides the ability to manage a group of IMSs (an IMSplex) from a single point of control (SPOC). That point of control is the OM API. You can write your own application using the OM API, use the IBM-provided TSO SPOC application, or write a REXX SPOC application that uses the REXX SPOC API. Using a SPOC application, you can:

- Issue commands to all the IMS subsystems in an IMSplex
- Display consolidated responses from those commands.
- Send a message to an IMS terminal that is connected to any IMS in the IMSplex using the BROADCAST command.

The use of a SPOC application is not required. The existing command interfaces for the WTOR, MTO, and E-MCS console continue to be supported.

For more information on the OM API or the REXX SPOC API, see *IMS Version 9: Common Service Layer Guide and Reference*.

An operator can use the TSO SPOC application to issue commands and receive responses to and from the various components of an IMSplex.

The TSO SPOC application does not have unique commands that are associated with it. The TSO SPOC supports the operator commands that are supported by the

OM API. Enter the command in the TSO SPOC command line and press Enter to run the command. The TSO SPOC supports the IMS type-1 commands and IMS type-2 commands.

Setting Up TSO SPOC

In order to use the TSO SPOC, the IMS distribution libraries added to the TSO user's environment. There are a couple of methods for setting up the TSO SPOC program.

1. One method is to edit the logon procedure to include the IMS distribution libraries or to issue ALLOCATE commands to make the data sets available to the TSO user.

   The TSOLIB command of TSO may be used to make the load module data set available to the TSO user. TSOLIB is a command that establishes a STEPLIB-like data set without having to modify the logon procedure.

   The data set allocated to ISPTABL is unique to the user. Because of the way ISPF uses tables, the same data set allocated to ISPTABL must also be allocated in the ISPTLIB concatenation and ahead of the IMS.SDFSTLIB data set. The data sets are shown in Table 31.

   Table 31. Data Sets for SPOC Setup

Usage	Data Set
TSOLIB command orSTEPLIB	IMS.SDFSRESL
FILE(ISPPLIB)	IMS.SDFSPLIB
FILE(ISPMLIB)	IMS,SDFSMLIB
FILE(ISPTLIB)	*user*.ISPTLIB IMS.SDFSTLIB
FILE(ISPTABL)	*user*.ISPTLIB
FILE(SYSPROC)	IMS.SDFSEXEC

   After the data sets are added to the your TSO environment, you can start the SPOC program by entering:

   ```
   TSO DFSSPOC
   ```

   in any ISPF command line, or by typing DFSSPOC in the ISPF option 6 command line.

2. A second method is to use the TSO ALTLIB command and ISPF's LIBDEF service. The DFSSPSRT exec provides an example of starting the DFSSPOC program using ALTLIB and LIBDEF.

   You can invoke DFSSPSRT from ISPF option 6:

   ```
   EXEC 'imsv810.SDFSEXEC(DFSSPSRT)' 'HLQ(imsv810)'
   ```

   where the data set name and the HLQ value uses your company's data set name prefix for the IMS distribution libraries.

   After setup is done using one of the methods described above, and the SPOC panel is displayed, set their execution preferences by doing the following:

   a. In the action bar, select Options.
   b. Select Preferences.
   c. Set the IMSplex name and wait interval fields in the preferences panel.
   d. Press Enter.

You can then enter commands at the TSO SPOC command line.

Using TSO SPOC

The TSO SPOC offers basic command entry with responses returned to the display area of the screen. On the initial entry to the TSO SPOC, no data is present. To issue a command, type in the command area and press Enter.

The command response from type-2 commands is in a tabular format. The columns are displayed in the order in which it is originally returned. You can sort the list by putting the cursor on the column title and pressing Enter, or by the use of the action bar.

The output from IMS type-1 commands is displayed in their existing format with the additional feature that one can scroll forward and back through the output if the output is greater than one page. Left and right scrolling is available for output wider than the screen width.

You can display or re-enter previously issued commands by using the action bar to select Display and Command Status. Other actions such as saving or printing the command response are also available from the action bar. Online help is available by pressing the F1 key.

Chapter 14. Data Sharing

When more than one IMS subsystem has concurrent access to an IMS database, those subsystems are sharing data, and you have a data-sharing environment. In a data-sharing environment, you manage multiple IMS subsystems, each running on different MVS systems in a sysplex. Such configurations require more sophisticated operations and recovery procedures. Descriptions of how to respond to failures in a single subsystem are not adequate.

This chapter describes the tasks associated with operations in a data-sharing environment, focusing on use of IMS commands. It relies on descriptions of standard operations elsewhere in this book, and it supports tasks similar to nonsharing operations tasks; however, it focuses specifically on the tasks you need to know about only when you operate in a data-sharing environment. These tasks are:

- Starting and stopping data sharing
  - Starting and stopping the IRLM
  - Starting and stopping database processing
- Controlling data sharing
  - Monitoring the system
  - Monitoring sysplex data sharing
  - Using DBRC
  - Changing database access intent
- Improving availability
  - Making online changes
  - Reorganizing databases
  - Making online image copies
- Improving database availability
- Maintaining normal operations
- Recovering from failures
  - Managing system logs
  - Setting up recovery procedures
  - Data sharing failures
  - Data sharing restart

This chapter assumes you already know how to install IMS subsystems, databases, DBRC, IRLM, and sysplex data sharing. For information about how to install data-sharing components, see *IMS Version 9: Installation Volume 1: Installation Verification* and *IMS Version 9: Installation Volume 2: System Definition and Tailoring*. For information about how to define, initialize, and tailor data-sharing components, see *IMS Version 9: Administration Guide: System*.

Starting and Stopping Data Sharing

Starting and stopping data-sharing operations involves:

- Starting and stopping the IRLM
- Starting and stopping database processing

Starting the IRLM

Start the IRLM from the system console of each participating system with the following MVS command: `START` *`irlmproc,parms`*. *irlmproc* is the unique name of the IRLM procedure, and *parms* represents one or more IRLM control parameters that override the command defaults.

Start the IRLM subsystems **before** starting the batch or online IMS; see “Starting the IRLM” on page 81.

If IRLM has not completed its initialization when the IMS subsystem tries to connect to it, IMS issues a `DFS039A` message that says the IRLM is not active, and asks the operator to reply to the message: retry, cancel, or dump.

- If you have not started the IRLM, issue the MVS `START` command, wait until the IRLM is active, and reply `RETRY`. IMS then tries to connect to the IRLM.
- If you have a problem coordinating startup, not merely starting the IRLM, reply `CANCEL` to cause the IMS subsystem to terminate with abend `U0039`.

  Example: Assume the following environment:
  – On processor J, an IMS subsystem (IMSJ) uses an IRLM (JRLM).
  – On processor K, an IMS subsystem (IMSK) uses an IRLM (KRLM).

  Next, assume the following scenario:
  1. A hardware error occurs on processor K.
  2. You try to restart IMSK on processor J to perform database backouts (but you do not change the IRLM name parameter, `IRLMNM=KRLM`).
  3. Message `DFS039A` indicates that KRLM is not active.

  Reply `CANCEL` in this case to immediately terminate the IMSK startup. Then, after changing `IRLMNM=` to `JRLM`, restart IMSK on processor J.
- If you reply `DUMP`, IMS terminates with abend `U0039` and a storage dump.

The MVS console operator must watch for messages originating from the IRLM. These messages have the three-letter prefix `DXR`.

Stopping the IRLM

You can terminate an IRLM normally only when all IMS subsystems that are using it have terminated. To stop an IRLM normally, use the MVS command `STOP` *`irlmproc`*. To stop an IRLM abnormally, use the MVS command `MODIFY` *`irlmproc`*`,ABEND,NODUMP`. To stop an IRLM abnormally and generate a dump, use the MVS command `MODIFY` *`irlmproc`*`,ABEND`.

Starting and Stopping Database Processing

In a data-sharing environment, IMS type-1 commands can act *locally*, that is, affect the local subsystem only, or can act *globally*, that is, affect all sharing subsystems in the sysplex. The `UPDATE` command can be routed to individual IMSs or to all IMSs in an IMSplex, using an automated operator program like the TSO SPOC.

An IMS command with the `LOCAL` keyword (or without the `GLOBAL` keyword) affects only the online subsystem on which you enter it. An IMS command with the `GLOBAL` keyword first affects the subsystem on which you enter it and then, if it is successful, affects other sharing subsystems in the sysplex.

When you enter a /START DB or UPDATE DB START(ACCESS) command, either locally or globally, DBRC makes sure that each registered database does not need recovery or backout. The IRLM communicates global commands only in a block-level data-sharing environment.

Table 32 summarizes the commands that affect data sharing when issued for registered databases. A command entered for a database affects full function transactions; a command entered for an area affects Fast Path transactions.

Table 32. Database Control Commands across Subsystems

Command	Action by DBRC	Effect in Local Subsystem	Effect in Sharing Subsystems
Local			
/RMCHANGE	Can change share level for subsystems using the database	None	None
/START AREA|DATABASE LOCAL or UPDATE	Ensures that the database or area does not need recovery or backout, and records database start	Allows transactions to access the database or area, and can change the access intent if you use the ACCESS= keyword	None
Global			
/DBDUMP AREA|DATABASE GLOBAL	Ensures that the database or area does not need recovery or backout, sets a return code for any unauthorized batch subsystems, records database close, and changes access level to read access	Prevents online updating access	Processes command as if entered locally
/DBRECOVERY AREA|DATABASE GLOBAL	Prohibits further authorization	Prevents allocation of database	Processes command as if entered locally
/STOP AREA|DATABASE GLOBAL	Records database status as stopped (restricts further authorization)	Stops transactions from accessing the database or area	Processes command as if entered locally
/STOP ADS	Records area data set as unavailable	Stops the area data set	Processes command as if entered locally
/START AREA|DATABASE GLOBAL	Ensures that the database or area does not need recovery or backout, resets the prohibit-further-authorization and read-only flags, records database start, and can change the access level	Allows transactions to access the database or area	Processes command as if entered locally

Example: You enter the /START command with the GLOBAL keyword on one subsystem and specify several database names, the IRLM transmits the command to other sharing subsystems, deleting the names of any databases that are invalid for the local system before it transmits the command, and all sharing subsystems process the command. In those online data-sharing systems, you see the DFS3334I message followed by the DFS3328I message, which tell you that the global /START command has started and has then completed. The messages include the database names that you include in your command.

If you omit the GLOBAL keyword (or specify the LOCAL keyword), the command applies only to the local online subsystem and does not affect access by any other IMS subsystem.

When you enter a /STOP or UPDATE command locally, no interaction occurs between IMS and DBRC.

Related Reading: For the format of these commands in a CICS environment, see *CICS-Supplied Transactions*.

Controlling Data Sharing

Controlling data sharing involves:

- Monitoring the system
- Using DBRC to control database allocations and access

Monitoring the System

To monitor data sharing, you obtain information on the status of the following: IRLM, IMS subsystems and databases, the RECON data set, and coupling facility structures.

Obtaining the Status of IRLM Activity

To display the status of an IRLM on either your system or on another connected system, enter the following MVS command:

```
MODIFY irlmproc,STATUS,irlmx
```

where *irlmproc* is the name of the procedure that you used to start the IRLM, and *irlmx* is the name of the IRLM whose status you want to display. You can use the ALLD keyword to display the names and status of every IMS identified to an IRLM in a data-sharing group. Or, you can use the ALLI keyword to display the names and status of every IRLM in a data-sharing group.

You can also trace IRLM activity. See “Tracing IRLM Activity” on page 114.

Related Reading: For a complete description of the commands for the IRLM, see *IMS Version 9: Command Reference*.

Displaying Components and Resources

Monitoring components and resources in a data-sharing environment requires the same kinds of procedures as in a non-sharing environment. For information about monitoring, see “Monitoring the System” on page 95.

Table 33 lists keywords for the /DISPLAY command that you can use to obtain information about various IMS resources.

Table 33. /DISPLAY Command Keywords That Provide Information about IMS Resources

Resources	/DISPLAY Command Keywords
Active control regions	ACTIVE REGION
Active jobs	ACTIVE
Programs, transactions, and conversations	CONVERSATION PROGRAM PSB STATUS PROGRAM STATUS TRANSACTION SYSID TRANSACTION TRANSACTION

Table 33. /DISPLAY Command Keywords That Provide Information about IMS Resources (continued)

Resources	/DISPLAY Command Keywords
Databases	DATABASE AREA STATUS DATABASE
Terminals, lines, links, and nodes	ACTIVE DC ASSIGNMENT LINE ASSIGNMENT LINK ASSIGNMENT NODE LINE LINK LTERM MASTER MSNAME NODE PTERM STATUS LINE STATUS LINK STATUS LTERM STATUS MSNAME STATUS NODE STATUS PTERM
External subsystems and connections to external subsystems	CCTL OASN SUBSYS SUBSYS
VTAM	TIMEOVER

For example, you can issue the `/DISPLAY DB` command after you issue a `/START DB` command to determine whether the database is started.

Table 34 lists keywords for the `QUERY` command that you can use to obtain information about various IMS resources.

Table 34. QUERY Command Keywords That Provide Information about IMS Resources

Resources	QUERY Command Keywords
Active control regions	MEMBER
Members in the IMSplex	IMSPLEX
Transactions	TRAN
Databases	DB AREA
Global online change status	OLC
Language environment (LE) runtime options	LE
Online reorganization (OLR) status	OLREORG
RM resource structure	STRUCTURE

For example, you can issue the `QUERY DB` command after you issue an `UPDATE DB START(ACCESS)` command to determine whether the database is started.

Related Reading: For detailed information about the `/DISPLAY` and `QUERY` commands, see *IMS Version 9: Command Reference*.

Monitoring Structures on a Coupling Facility

The following MVS operator commands are especially useful in monitoring structure activity:

- `DISPLAY XCF,STRUCTURE`
- `DISPLAY XCF,STRUCTURE,STRNAME=`

These commands let you look at structures on a coupling facility to determine resource status and, for failures, gather information for problem determination.

For detailed information on these commands, see the appropriate MVS publication.

DISPLAY XCF,STRUCTURE Command: Use this command to display the status of structures defined in your active policy.

Example: Figure 14 shows an example of output from this command.

```
IXC359I  11.09.26  DISPLAY XCF 376
STRNAME           ALLOCATION TIME  STATUS
CF01             02/17/96 17:03:49 ALLOCATED
CF02                --       --    NOT ALLOCATED
CF03                --       --    NOT ALLOCATED
CF04                --       --    NOT ALLOCATED
OSAMSESXI        02/17/96 17:02:54 ALLOCATED
                                   REBUILDING
                                   REBUILD PHASE: QUIESCE
VSAMSESXI        02/17/96 17:03:03 ALLOCATED
```

Figure 14. Output from a DISPLAY XCF,STRUCTURE Command

Observe the following in Figure 14:

`STRNAME`
Is the name of a structure.

`ALLOCATION TIME`
Is a timestamp indicating when the structure was allocated in the coupling facility.

`STATUS`
Is the current status of the structure. A structure can be in a number of different states, such as ALLOCATED, NOT ALLOCATED, or ALLOCATED REBUILDING.

The displayed information can help you determine whether the appropriate IRLM, OSAM, VSAM, shared MSGQ, and shared EMHQ structures have been defined and allocated. If not, check that the structure names on your `CFNAMES` control statement match the structure names that are displayed. If they do not match, change the names in either the coupling facility resource manager (CFRM) policy or the `CFNAMES` control statement.

In Figure 14, six structures are defined (CF01-CF04, OSAMSESXI, and VSAMSESXI). Three of the structures are allocated (CF01, OSAMSESXI, and VSAMSESXI). The status of the OSAM structure is that it is currently being rebuilt after a structure failure.

DISPLAY XCF,STRUCTURE,STRNAME= Command: Use this command to display detailed information about a specific structure. The three structures of importance to IMS are the IRLM, OSAM, VSAM, shared MSGQ, and shared EMHQ structures.

Example: Figure 15 shows an example of output from this command, specifying an OSAM structure named OSAMSESXI.

```
 IXC360I  17.30.46  DISPLAY XCF 677
 STRNAME: OSAMSESXI
  STATUS: ALLOCATED
   POLICY SIZE    : 2048 K
   PREFERENCE LIST: CF02     CF01
   EXCLUSION LIST IS EMPTY

  ACTIVE STRUCTURE
  ----------------
   ALLOCATION TIME: 02/17/96 17:02:54
   CFNAME         : CF02
   COUPLING FACILITY: ND02...
                      PARTITION: 0    CPCID: 00
   ACTUAL SIZE    : 2048 K
   STORAGE INCREMENT SIZE: 256 K
   VERSION        : A8DAC970 15774B04
   DISPOSITION    : DELETE
   ACCESS TIME    : 0
   MAX CONNECTIONS: 32
   # CONNECTIONS  : 5

  CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
  ---------------- -- -------- -------- -------- ---- ------
  DLI11            04 00040001 MVS1     DLI11    0031 ACTIVE
  DLI12            05 00050001 MVS1     DLI12    0033 ACTIVE
  IMS1             01 00010013 MVS1     DLIOCSA8 0036 ACTIVE
  IMS2             02 00020012 MVS2     DLIOCSB8 0035 ACTIVE
  IMS3             03 00030011 MVS3     DLIOCSC8 0038 ACTIVE
```

Figure 15. Output for DISPLAY XCF,STRUCTURE,STRNAME= Command

There are three parts to the output. The first part shows the status of the structure and information about the active policy. The following information from the display is especially useful:

STRNAME
Is the name of the structure, as specified in the command.

STATUS
Is the current status of the structure. A structure can be in a number of different states, such as ALLOCATED, NOT ALLOCATED, or ALLOCATED REBUILDING.

POLICY SIZE
Is the size of the structure as specified in the policy.

PREFERENCE LIST
Shows the coupling facility on which MVS allocates the structure. MVS tries to allocate the structure on the first coupling facility listed (CF02 in Figure 15), then the second one (CF01), and so on.

The second part of the output shows more detailed status information for the structure.

ALLOCATION TIME
Is the time the structure was allocated.

CFNAME
Is the name of the coupling facility on which the structure was allocated.

COUPLING FACILITY
Is the name of the coupling facility on which the structure resides.

ACTUAL SIZE
Is the actual size of the structure.

DISPOSITION
Is what the system will do with the structure in a failure situation. The IRLM structure always has a disposition of KEEP. The OSAM and VSAM structures always have a disposition of DELETE.

MAX CONNECTIONS
Is the maximum number of connections that can be made to the OSAM, VSAM, IRLM, shared MSGQ, or shared EMHQ structure. The maximum is 32 connections for any structure.

# CONNECTIONS
Is the current number of IMS systems connected to the structure.

Other information in the second part of the output is not needed for monitoring and problem determination for IMS sysplex data sharing.

The third part of the output shows which MVS systems are connected to the structure and gives information about each use.

CONNECTION NAME
Is the name of a connection. MVS will repeat the connection message text in a table to report all connections. If there are no connections, no connection table is displayed.

ID Is the connection identifier.

VERSION
Is the version number of the MVS system that is connected.

SYSNAME
Is the name of the MVS system that is connected.

JOBNAME
Is the name of the job associated with the connection.

ASID
Is the ID of the address space associated with the connection.

STATE
Is the status of the structure, which can be one of the following:

FAILED PERSISTENT
The failed persistent status occurs if the disposition of a structure is KEEP and connection to the structure is lost.

DISCONNECTING
IMS is in the process of disconnecting from the structure.

FAILING
IMS is in the process of abnormally terminating.

`ACTIVE`
IMS is connected.

`ACTIVE &`
IMS is connected, but has physically lost connectivity to the structure.

`ACTIVE OLD`
The structure is being rebuilt. IMS is connected to the old structure.

`ACTIVE &OLD`
The structure is being rebuilt. IMS is connected to the old structure, but has physically lost connectivity to it.

`ACTIVE NEW,OLD`
The structure is being rebuilt. IMS is connected to both the old and the new structure.

`ACTIVE NEW,&OLD`
The structure is being rebuilt. IMS is connected to both the old and the new structure, but has physically lost connectivity to the old structure.

`ACTIVE &NEW,OLD`
The structure is being rebuilt. IMS is connected to both the old and the new structure, but has physically lost connectivity to the new structure.

`ACTIVE &NEW,&OLD`
The structure is being rebuilt. IMS is connected to both the old and the new structure, but has physically lost connectivity to both structures.

Using DBRC to Control Access to Data

DBRC allows you to control access to data by IMS subsystems that participate in data sharing. Using DBRC, you can modify, initiate, and delete the current status indicators in the RECON data set to change:

- The access intent of online IMS subsystems
- The share level of registered databases

Your data sharing environment depends on the status of the databases and subsystems indicated in the RECON data set.

| You can modify the access intent indicator using a form of the `/START` or `UPDATE` command. For a description of these commands, see "Changing Database Access Intent" on page 218.

You can modify the share level indicator using a form of one of the DBRC online change commands, `/RMCHANGE`.

Online DBRC Commands

The `/RMCHANGE` command is one of a set of IMS commands that control DBRC utility functions; IMS responds to them by issuing corresponding commands directly to DBRC. Table 35 shows these commands and their recommended uses.

Related Reading: For full descriptions of these commands and their functions, see *IMS Version 9: DBRC Guide and Reference*.

Table 35. DBRC Commands and Functions

Command	Recommended Use
/RMCHANGE	Alter a database sharing level Prevent other subsystem authorization Set or remove backout-required status

Table 35. DBRC Commands and Functions (continued)

Command	Recommended Use
/RMDELETE	Remove database record and associated status Delete subsystem with unauthorized databases Remove area data set (ADS) record and associated status
/RMGENJCL	Generate online recovery jobs
/RMINIT	Register a new database Start control of an area data set
/RMLIST	Obtain recovery and authorization information for a database or ADS
/RMNOTIFY	Reset a system status

Example: If you want to change the share level of a registered database (named ORDERDB) from intraprocessor block-level data sharing (share-level 2) to interprocessor block-level data sharing (share-level 3) to allow data sharing with another IMS subsystem in another processor, enter:

```
/RMCHANGE DBRC='DB DBD(ORDERDB) SHARELVL(3)'
```

DBRC replies with its corresponding command input (`CHANGE.DB DBD(ORDERDB) SHARELVL(3)`) and a series of DBRC messages.

Do not enter the `/RMCHANGE`, `/RMDELETE`, or any command that alters the status of database records in the RECON data set while a job accesses the database. Stop the database using a global `/DBRECOVERY DB` or `UPDATE DB STOP(ACCESS)` command before modifying any RECON record for that database. Otherwise, IMS rejects the command with an error.

Denial of Authorization

The following occurs if DBRC (in a data sharing environment) responds to authorization requests but fails to obtain authorization for a program:

- You receive message `DFS047A` identifying the database.
- IMS schedules the program's PSB without database access. IMS abnormally terminates a BMP or MPP with abend `U3303` only if it tries to access the database.
- IMS abnormally terminates batch and utility regions with abend `U0047`.

Changing Database Access Intent

Use one of the following commands to change the access intent that you declare during system definition:

- `/START DATABASE` *dbx* `ACCESS=`*xx*
- `UPDATE DB START(ACCESS) SET(ACCTYPE(BRWS,EXCL,READ,UPD))`

where *dbx* is the database name and *xx* is the new access intent.

For the `/START DATABASE` *dbx* `ACCESS=`*xx* command, values for *xx* are:

EX Exclusive use

UP Update access

RD Read access

RO Read-only access

This command is local; it affects only the subsystem on which you enter it. The `GLOBAL` keyword is not valid with the `ACCESS=` keyword. If you need to change the access level for a shared database across the sysplex, you must enter this command on each subsystem that shares the database.

The `UPDATE DB START(ACCESS) SET(ACCTYPE(BRWS,EXCL,READ,UPD))` command can be either local or global.

In order to change the access intent for a DEDB, you must stop all PSBs that access any of the areas in the DEDB. You might also have to stop regions that have wait-for-input (WFI) transactions scheduled for the DEDB.

Improving Database Availability

Improving database availability in a data-sharing environment involves:

- Controlling online change
- Reorganizing databases
- Making database image copies

Controlling Online Change

If you modify a shared database online in a sharing environment, you must take these precautions:

- You must coordinate the sequence of `/MODIFY` commands across all sharing subsystems:
  1. Stop the databases you want to modify in all IMS online subsystems using that database.
  2. Perform the online change on all sharing subsystems
  3. Restart the shared databases only when the `/MODIFY` command sequences are complete in all affected subsystems.
- If any of the data-sharing subsystems also physically share any of the ACBLIB, FORMAT, or MODBLKS libraries, you must ensure that these subsystems use the same designated active library, for example, IMS.ACBLIBA.

  Each system keeps track of its active libraries in the IMS.MODSTAT data set. Performing an online change in only one subsystem could result in the individual subsystems' using different control blocks, which can cause unpredictable errors in a data-sharing environment.

  Also, if any library is shared, ensure that the Online Change utility (DFSUOCU0) updates only inactive libraries.

Reorganizing Databases

While you reorganize a database (using the database reorganization utilities), you must protect it from changes by subsystems that are authorized to use it, unless you are using HALDB Online Reorganization (OLR). You have three ways to protect the database:

- Issue the `/DBRECOVERY DB GLOBAL` or `UPDATE DB STOP(ACCESS)` command to close the database on all subsystems. After the database reorganization is complete, tell the MTO of each subsystem that the database is available.
- Issue the DBRC `CHANGE.DB NOAUTH` command to specify that the database is not authorized to participate in data sharing. After the reorganization is complete, issue the `CHANGE.DB AUTH` command to resume data sharing.
- Devise an MTO procedure to coordinate application activities in sharing subsystems. This procedure would direct the MTOs to stop applications that might be active and using the database. After the reorganization is complete, the MTOs can restart applications in the appropriate online IMS subsystems.

The reorganization utilities communicate with DBRC. In response, DBRC does not allow further processing for the database, even after the reorganization, until you make an image copy of the database or recovery group.

The reorganization utilities might not execute if they fail to obtain authorization from DBRC for the database. DBRC denies authorization if it has authorized another subsystem to update the database (which is why you need to issue the `/DBRECOVERY DB`, `UPDATE DB STOP(ACCESS)`, or `CHANGE.DB` commands first). If you receive message `DFS044I` saying that DBRC is required for the execution of the utility, and the utility abends, wait for the subsystem to terminate or, if it is an online IMS, issue a `/DBRECOVERY DB` or `UPDATE DB STOP(ACCESS)` command.

Related Reading: For a sample operational scenario dealing with database recovery in a database-level sharing environment, see “Reorganizing a Database” on page 222. For information on running the reorganization utilities, see *IMS Version 9: Utilities Reference: Database and Transaction Manager*.

Making an Online Database Image Copy

When you schedule the Online Database Image Copy utility (DFSUICP0) as a BMP, DBRC manages the requirement for restrictive access. You must, however, ensure that no IMS online subsystems, online or batch, other than the IMS online subsystem running the Image Copy BMP, is authorized to update the database.

For a sample operational scenario supporting image copy, see “Making an Image Copy” on page 224.

Maintaining Normal Operations

Data sharing requires additions to your normal operating procedures. This section presents sample operational scenarios that show how to perform routine adjustments and maintenance in a database-level data-sharing environment:

- Scheduling a batch subsystem to update a database
- Reinstating update capability to an online subsystem after a batch updater terminates
- Transferring update capability from one sharing online subsystem to another
- Reorganizing a shared database
- Making an image copy of a database with exclusive control
- Making an image copy of a shared database
- Moving from database-level sharing to exclusive control by one subsystem

Recommendation: Use an External Time Reference (ETR) device and set it to use Universal Coordinated Time. You can set the local time for each data-sharing subsystem separately as necessary. See *OS/390 MVS Initialization and Tuning Reference* for information on using an ETR.

Scheduling a Batch Updater

Given the following initial and target configurations, use this scenario to schedule a batch updater:

- Initial configuration
  - Database x (*dbx*) is registered at share level 1.
  - *dbx* is authorized for update access to an online IMS subsystem (IMSA).
- Target configuration

  - *dbx* is authorized for update access to a batch IMS subsystem (IMSB).
  - *dbx* is authorized for read-only access to IMSA.

Follow these steps to schedule a batch updater:

1. Prevent other subsystems from obtaining access to *dbx* with the following command:

   ```
   CHANGE.DB DBD (dbx) NOAUTH
   ```

2. Suspend IMSA's activity on *dbx*, close *dbx*, and release IMSA's authorization with one of the following commands:
   - `/DBRECOVERY DB dbx NOFEOV`
   - `UPDATE DB STOP(ACCESS) OPTION(FEOV)`

   You can use the /DBRECOVERY or UPDATE command for a DEDB area by specifying AREA area instead of DB.

3. Allow read-only processing from IMSA with one of the following commands:
   - `/START DATABASE dbx ACCESS=RO`
   - `UPDATE DB NAME dbx START(ACCESS) SET(ACCTYPE(READ))`

4. Allow IMSB to initialize with the following command:

   ```
   CHANGE.DB DBD (dbx) AUTH
   ```

5. Start IMSB with update access to *dbx*.

   Related Reading: For information on how to start an IMS subsystem, see Chapter 8, "Starting or Restarting IMS," on page 75. For information about how to specify update access, see *IMS Version 9: Administration Guide: System*.

Reinstating an Online Updater

Given the following initial and target configurations, use this scenario to restore the initial configuration of the scenario described in "Scheduling a Batch Updater" on page 220:

- Initial configuration
  - *dbx* is authorized for read-only access to an online IMS subsystem (IMSA).
  - A batch IMS subsystem has been terminated.
- Target configuration
  - *dbx* is registered at share level 1.
  - *dbx* is authorized for update access to IMSA.

Follow these steps to reinstate an online updater:

1. Prevent other subsystems from obtaining access to *dbx* with the following command:

   ```
   CHANGE.DB DBD (dbx) NOAUTH
   ```

   When you force IMSA to release its authorization on *dbx*, another subsystem could obtain authorization to process it. The DBRC CHANGE.DB command with the NOAUTH keyword indicates (in the database record in the RECON data set) that the database is unavailable for processing.

2. Terminate IMSB.

3. Suspend IMSA's activity on *dbx*, close *dbx*, and release IMSA's authorization with one of the following commands:
   - `/DBRECOVERY DB dbx NOFEOV`
   - `UPDATE DB STOP(ACCESS) OPTION(FEOV)`

4. Allow update processing for IMSA with one of the following commands:
   - `/START DB dbx ACCESS=UP`
   - `UPDATE DB START(ACCESS)`
5. Allow access or update in IMSA with the following command:

   ```
   CHANGE.DB DB dbx AUTH
   ```

Transferring Update Capability

Given the following initial and target configurations, use this scenario to transfer update capability from one sharing subsystem to another:

- Initial configuration
  - *dbx* is authorized for update access to an online IMS subsystem (IMSA).
  - *dbx* is authorized for read-only access to another online IMS subsystem (IMSB).
- Target configuration
  - *dbx* is authorized for read-only access to IMSA.
  - *dbx* is authorized for update access to IMSB.

Follow these steps to transfer update compatibility:

1. Prevent other subsystems from obtaining access to *dbx* with the following command:

   ```
   CHANGE.DB DBD (dbx) NOAUTH
   ```
2. Suspend activity and close *dbx* in both subsystems with one of the following commands:
   - `/DBRECOVERY DB dbx NOFEOV`
   - `UPDATE DB STOP(ACCESS)`
3. Allow processing in both subsystems. Issue one of the following commands on IMSA:
   - `/START DB dbx ACCESS=RO`
   - `UPDATE DB START(ACCESS)`

   Then issue one of the following commands on IMSB:
   - `/START DB dbx ACCESS=UP`
   - `UPDATE DB START(ACCESS)`
4. Allow access or update in both subsystems with the following command:

   ```
   CHANGE.DB DBD(dbx) AUTH
   ```

   Although IMSA and IMSB now have compatible database authorizations, the database is not protected from other subsystems until IMSA and IMSB schedule an application program with a PSB that is sensitive to the database. In order to lock out any other updaters, therefore, IMSB must be the first subsystem to obtain the authorization to process the database with update access.

Reorganizing a Database

Given the following initial and target configurations, use this scenario to reorganize a database:

- Initial configuration
  - *dbx* is registered for database-level sharing.
  - *dbx* is authorized for update access to an online IMS subsystem (IMSA).
- Target configuration

- *dbx* is successfully reorganized.

Follow these steps to reorganize a database:

1. Prevent other subsystems from obtaining access with the following command:

   ```
   CHANGE.DB DBD(dbx) NOAUTH
   ```

2. Start reorganization jobs.

   Run a set of reorganization jobs, such as Scan, Unload, Reload, or Prefix Update, in a protected environment.

   You must run these reorganization utilities without interference from other subsystems. The DBRC `CHANGE.DB NOAUTH` command prohibits authorizations to the database for online and batch subsystems, except for all reorganization utilities.

3. Authorize the Online Database Image Copy utility (DFSUICP0) initialization with the following command:

   ```
   CHANGE.DB DBD(dbx) AUTH
   ```

   This command allows other batch or online subsystems to obtain access to *dbx*.

Related Reading: For a discussion of ways to reorganize a database, see "Reorganizing Databases" on page 219.

Making an Image Copy with Exclusive Control

Given the following initial and target configurations, use this scenario to make an image copy of a database data set with exclusive control:

- Initial configuration
  - *dbx* is registered for database-level sharing.
  - *dbx* is authorized for update access to an online IMS subsystem (IMSA).
  - *dbx* is authorized for read-only access to another online IMS subsystem (IMSB).
- Target configuration
  - An image copy of database *dbx* (data set *y*).

Follow these steps to make an image copy with exclusive control:

1. Prevent other subsystems from obtaining access with the following command:

   ```
   CHANGE.DBDS DBD(dbx) DDN(y) ICON
   ```

   Use the DBRC `CHANGE.DBDS` command with the `ICON` keyword to indicate (in the RECON data set) that an image copy is needed. This indication prevents any subsystem except the Image Copy utility from obtaining access to the database.

2. Suspend subsystem activity on *dbx* and close *dbx* in all sharing subsystems with one of the following commands:
   - `/DBRECOVERY DB dbx NOFEOV`
   - `UPDATE DB STOP(ACCESS) OPTION(FEOV)`

3. Run the Online Database Image Copy utility (DFSUICP0).

   When this utility completes successfully, it removes the indication in the RECON data set that an image copy is needed, so that DBRC can authorize subsystems to access the database.

4. Resume subsystem activity on *dbx* with one of the following commands:
   - `/START DB dbx`
   - `UPDATE DB START(ACCESS)`

Making an Image Copy

Given the following initial and target configurations, use this scenario to make an image copy in a database-level sharing environment:

- Initial configuration
  - *dbx* is registered for database-level sharing.
  - *dbx* is authorized for update access to an online IMS subsystem (IMSA).
  - *dbx* is authorized for read-only access to a batch IMS subsystem (IMSB).
- Target configuration
  - The Online Database Image Copy utility (DFSUICP0) on IMSA is authorized for read access.
  - *dbx* is authorized for read access to IMSA.
  - *dbx* is authorized for read-only access to IMSB.

Because the Online Database Image Copy utility (DFSUICP0) always requests read access, it can share the database only with subsystems that request a read-only access. If the other subsystems need to update the database while you take an image copy, use the Concurrent Image Copy option.

Follow these steps to make an image copy:

1. Prevent all subsystems from obtaining update access with the following command:

   ```
   CHANGE.DB DBD(dbx) READON
   ```

   This command indicates (in the RECON data set) that the database cannot be accessed by subsystems with update intent.
2. Suspend IMSA update activity on *dbx* and lower authorization to **read access** with one of the following commands:
   - `/DBDUMP DB dbx NOFEOV`
   - `UPDATE DB STOP(UPDATES) OPTION(FEOV)`
3. Run the Online Database Image Copy utility (DFSUICP0).
4. Resume update activity on IMSA with one of the following commands:

   -
     ```
     CHANGE.DB DBD(dbx) READOFF
     /START DATABASE dbx ACCESS=UP
     ```

   -
     ```
     CHANGE.DB DBD(dbx) READOFF
     UPDATE DB NAME(dbx) START(ACCESS) SET(ACCTYPE(UPD))
     ```

   The `CHANGE.DB` command with the `READOFF` keyword indicates (in the RECON data set) that the database can be accessed by subsystems with update intent.

Lowering the Share Level of a Database

Given the following initial and target configurations, use this scenario to lower the share level of a database:

- Initial configuration
  - *dbx* is authorized for update access to an online IMS subsystem (IMSA).
  - *dbx* is authorized for read-only access to another online IMS subsystem (IMSB).
- Target configuration
  - *dbx* is authorized for exclusive control by IMSA.

Follow these steps to lower the share level:

1. Prevent all subsystems from obtaining access with the following command:
   ```
   CHANGE.DB DBD(dbx) NOAUTH
   ```
2. Suspend activity on *dbx* and close *dbx* in both subsystems with one of the following commands:
   - `/DBRECOVERY DB dbx NOFEOV`
   - `UPDATE DB STOP(ACCESS) OPTION(FEOV)`
3. Allow processing in IMSA with the following command:
   ```
   CHANGE.DB DBD(dbx) AUTH
   /START DATABASE dbx ACCESS=EX
   /START PGM
   /START TRAN
   ```

   The database is not protected from other subsystems until IMSA schedules an application program with a PSB that is sensitive to the database. In order to lock out all other subsystems, therefore, IMSA must be the first subsystem to obtain the authorization to process the database with exclusive access.

   IMSA accepts the change of access intent only if the target database is not authorized.

You can also achieve the same results by modifying the sharing level of the database in the RECON data set instead of changing the access intent of IMSA. Omit the `/START DATABASE` command with the `ACCESS=` keyword and perform the following step before step 3:

Request DBRC to enforce exclusive control:

```
CHANGE.DB DBD(dbx) SHARELVL(0)
```

In this case, DBRC grants database authorization at the requested level, and IMSA owns the database exclusively.

Recovering from Failures

This section provides information to help you plan for and recover from failures in a data-sharing environment. Specifically, it describes:

- Managing System Logs
- Planning Recovery Procedures
- Recovery Facilities
- Recovery without DBRC
- Restart after IMS Failure
- Restart after DBRC Failure
- Recovery Involving IRLM Configurations
- Recovering from sysplex Data Sharing Failures

Managing System Logs

Recovering a shared database requires log data from each sharing subsystem. A DBRC `GENJCL.RECON` command or a Database Recovery utility execution using DBRC will fail if logs need to be merged for the database. You can use the merge function of the IMS Database Change Accumulation utility (DFSUCUM0) to do merge the log data from all sharing subsystems.

You can run the Database Change Accumulation utility (DFSUCUM0) as often as your workload requires **without** first establishing a complete set of log volumes.

Doing so allows you to accumulate change records on a timely basis, keeping the log input to a reasonable number of volumes, without taking databases offline or bringing down IMS. Doing so can, however, increase the size of the change accumulation data set.

IMS stores records from these partial subsets of log volumes on the change accumulation data set as “spill records”. These records retain their individual update characteristics.

Before running the Database Recovery utility (DFSURDB0), you must run the Database Change Accumulation utility, specifying a complete set of log volumes for the DBDS that you need to recover. If you run the Database Change Accumulation utility with an incomplete set of logs, specify the change accumulation output data set as input to your next Database Change Accumulation run. If a change accumulation data set represents an incomplete set of logs for the DBDS that you need to recover, the data set is not valid for recovery of that DBDS.

The DBRC `LIST.CAGRP` command displays whether the set of log volumes for each DBDS of the change accumulation group is complete or incomplete.

You can run DFSUCUM0 with a valid set of log volumes at any time in order to minimize the size of the change accumulation data set. To establish a valid set of logs in a data-sharing complex, use the following procedure:

1. Suspending operation of all batch subsystems
2. Using the `/DBRECOVERY DB` or `UPDATE DB STOP(ACCESS)` command to terminate database access for all but one online subsystem
3. Forcing a log switch on the remaining online subsystem

Planning Recovery Procedures

Recovery in a data-sharing environment is similar to standard IMS recovery, in that it involves these primary tasks:

- Setting up the mechanisms of recovery (logging, taking checkpoints, keeping records)
- Setting up the operational procedures to be followed in the event of situations that require recovery

The logging and checkpoint mechanisms of online IMS subsystems in a nonsharing environment are also active in a data-sharing environment. These include:

- System log data sets and the use of the WADS and restart data sets
- Program, system, and message-queue checkpoints
- Making image copies of databases

The primary difference between nonsharing and data-sharing environments is in their degree of reliance on DBRC. DBRC helps control the data-sharing environment; it does not merely keep records.

As in a single IMS subsystem, a failure in a data-sharing environment may require offline (utility-type) recovery as well as restart. After a failure in a data-sharing environment, however, you must consider and perform recovery actions not only in the failing system, but also in other sharing systems. In addition, each IRLM, DBRC, and coupling facility structure in the environment introduces another potential point of failure and another component that you might need to include in a recovery and restart procedure.

Recovery for the message queue data sets, (in a nonshared-queues environment) log data sets, and system data sets are no different in a data-sharing environment:

- To recover the log, use the Log Recovery utility (DFSULTR0).
- To recover the message queue data set, request at IMS restart that IMS rebuild the queues. You need to supply the system log data sets since the last system checkpoint that recorded the queue contents (SNAPQ, DUMPQ, or PURGE).
- To recover a system data set other than the message queue data set, update the latest image copy. You must recover the RECON data set for data sharing to continue.

Related Reading: For more information about non-sharing recovery procedures, see Chapter 12, “Recovering from IMS Failures,” on page 153.

Recovery Facilities

The following recovery facilities in a data-sharing environment are not the same as in a nonsharing environment:

- Dynamic backout
- Batch backout
- Forward recovery

Dynamic Backout

As in a nonsharing environment, an online IMS subsystem in a data-sharing environment dynamically backs out an application program’s uncommitted database changes and discards its uncommitted output messages if the program fails or requests backout with a rollback call. In a data-sharing environment, however, IRLM locks and RECON status indicators ensure integrity by protecting uncommitted changes from sharing subsystems. After an application program failure, operation of the system and other programs continues uninterrupted.

If the application program is a batch updater in a block-level data-sharing environment, IMS dynamically backs out changes in any of the following situations:

- When the program issues a rollback call (ROLL, ROLB, or ROLS).
- When the batch region uses system logging allocated to DASD, you specify BKO=Y as an execution parameter, and the program issues a ROLB or ROLS call or a DL/I-detected error occurs.
- When some types of resource shortages and deadlocks involving two or more batch update programs occur, IMS performs dynamic backout for one or more of the batch programs; all continue executing, provided that the involved regions use DASD logging and you specify BKO=Y.

In these situations, you are not involved in recovery actions.

For either online or batch subsystems, if the database is shared at the database level, other programs might read uncommitted data before the database is dynamically backed out. Backout affects only the updating program’s data in the database—not the output of any other program that used this uncommitted data. This is a consequence of the nature of PROCOPT=GO processing, rather than of the nature of database backout in a data-sharing environment.

If a dynamic backout for a database fails, you must back out the database using batch backout. Then, you must restart that database using a /START DB or UPDATE DB START(ACCESS) command.

Batch Backout

If the failing program is a batch updater involved in block-level data sharing, IMS does not always automatically back out its uncommitted database changes. Just as in a nonsharing environment, you must recover the database if any of the following is true:

- The batch region does not use DASD logging
- You do not specify `BKO=Y` as an execution parameter
- The failure is not a DL/I-detected error

In a block-level-data-sharing environment, however, you must back out database changes of all batch update programs using the Batch Backout utility (DFSBBO00).

During block-level sharing, if a program deadlock occurs involving only batch update jobs, IMS abnormally terminates one of these jobs with abend `U0777`. If IMS can dynamically backout changes, other programs continue processing. Otherwise, IMS abnormally terminates the other programs with abend `U3303`.

You must recover all of the batch jobs that did not have dynamic backout by executing the Batch Backout utility for each of them. Because the locks held by these batch jobs could affect access of other batch jobs or online subsystems, you should begin batch backouts promptly.

Restrictions: For recovery of batch subsystems that participate in block-level data sharing:

- The Batch Backout utility requires an IRLM for its execution.
- The Batch Backout utility backs out to the last successful checkpoint.

Forward Recovery

Recovery of a database in a data-sharing environment is similar to recovery in a nonsharing environment. In both environments, use the Database Recovery utility (DFSURDB0) and give it the most recent image copy of the database, along with all pertinent log data sets used since you made that image copy.

Block-level data sharing, however, may require one additional step. Because more than one subsystem might update the database, you need the log from each such subsystem to reconstruct the database. Furthermore, because these updates are most likely to be concurrent, you cannot submit those logs sequentially to the Database Recovery utility. Instead, you must first merge them by running the Database Change Accumulation utility (DFSUCUM0). Be sure to process all pertinent logs.

Even if you run change accumulation regularly, you must run the utility again before recovering the database so you merge the most recent logs with earlier logs.

However, you do not need to establish a valid set of logs for the Database Change Accumulation utility (DFSUCUM0). You can run DFSUCUM0 at any time without suspending operation of batch subsystems, terminating database access for all but one online subsystem, or forcing a log switch on the remaining online subsystem. See “Managing System Logs” on page 225.

Recovery without DBRC

If you perform any recovery-related actions when DBRC is not running, such as making database image copies, problems could arise because DBRC would be unaware of changes in status. You must, therefore, specifically inform DBRC of such changes. DBRC offers several commands for this purpose; see “Online DBRC Commands” on page 217.

Because restart might be required, you should, for example, use the Utility Control Facility (UCF) without DBRC being active. But, if you use UCF for any recovery-related work, you must specifically notify DBRC of status changes afterward.

Restart after IMS Failure

Restart an IMS subsystem in a data-sharing environment in the same way as in a nonsharing environment:

- Restart IMS normally, using the `/NRESTART` command, if you were able to shut the subsystem down normally.
- Emergency restart IMS, using the `/ERESTART` command, if the subsystem failed.

In a data-sharing environment, however, consider that if the associated IRLM also stops or fails, you must restart that IRLM before you start IMS in a block-level data-sharing environment. See “Starting the IRLM” on page 210.

Restart after DBRC Failure

Because DBRC runs under the control of IMS, if IMS abnormally terminates, IMS tries to release DBRC buffers, and to close the system log. After you correct the DBRC problem, restart IMS using the `/ERESTART` command.

Recovery Involving IRLM

Failure in one part of a data-sharing configuration does not usually cause the other parts to fail. For example:

- If one MVS system fails because of a hardware problem or an abnormal termination of the operating system, the other systems—with their IRLMs and other IMS subsystems—continue to run. In this case, the surviving IRLMs are notified of the failure by MVS and then provide the necessary retained lock protection for locks owned by the failing member.
- If an IRLM fails, its associated IMS online subsystems continue to run while all active application programs receive `U3303` abends, and IMS suspends transaction scheduling, allowing backouts to complete. Batch IMS subsystems terminate to protect database integrity. The surviving IRLMs are notified of the failure by MVS and then provide the necessary retained lock protection for locks owned by the IMS on the failing IRLM.

  If an IRLM fails in an IMS DBCTL environment, in-doubt threads receive a `U3303` abend on their next DL/I calls. The in-doubt threads remain active until phase 2 of sync-point processing. After processing phase 2 commit or abort, the thread is terminated. The IMS DBCTL subsystem tries to reconnect to IRLM after all in-doubt threads are terminated.
- If an IMS subsystem fails, its associated IRLM and other IMS subsystems continue to run, with locks owned by the failing IMS protected by retained locks created by its IRLM.
- If DBRC fails, its IMS subsystem abnormally terminates, but the associated IRLM and the other IMS subsystems and their IRLMs continue to run.

Transaction Recovery in Online IMS Subsystems

IMS automatically backs out transactions if the IMS online subsystem determines that transaction processing cannot proceed because of problems directly or indirectly to do with data sharing, or because the resource is unavailable, and if you specified SERIAL=NO in the TRANSACT system definition macro (or accepted it as the default).

If you specify SERIAL=YES in the TRANSACT system definition macro, IMS puts the transaction back on the front of the transaction queue to be processed first-in-first-out, and stops the transaction (USTOP state).

If the transactions do not use the DL/I INIT STATUS GROUPA call, IMS places transactions on either the suspend queue or the transaction queue when:

- A transaction requests a lock held by a subsystem identified to a failed IRLM.
- A transaction requests a lock held by a failed subsystem.
- The transaction attempts to access a database at a level higher than that for which the subsystem is authorized.
- A transaction requires a database that is stopped or locked.
- The MTO enters a /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command.

IMS sends message DFS3324I, saying that the transaction is suspended, for each individual transaction that is suspended.

IMS saves the suspend queue across any restart, except a cold start, and all transactions on the suspend queue are eligible for reprocessing. When IMS reprocess the transactions, IMS releases them to the message queues and schedules them in the normal way.

Several conditions prompt IMS to attempt to reprocess these transactions:

- A restart of the online IMS system, including after an XRF takeover
- A communication from another IMS subsystem performing restart or from a batch subsystem that has completed backout
- A reconnection of an IMS online system to an IRLM
- The MTO command, /DEQUEUE SUSPEND
- The MTO commands, /START DB or UPDATE DB START(ACCESS) to start the database

IMS issues message DFS3336I when it attempts to reprocess suspended transactions.

Restart with a Single Failing IRLM

If the only IRLM in a data-sharing configuration fails:

- The associated IMS online subsystems suspend all transaction scheduling.
- IMS performs backout for affected transactions.
- IMS places the transactions on a suspend queue or transaction queue, depending on whether you specified SERIAL=NO or SERIAL=YES in the TRANSACT macro (see “Transaction Recovery in Online IMS Subsystems” on page 229).
- IMS stops shared databases.

IMS issues DFS2012I messages for each database that has been stopped automatically because of the failure. IMS abnormally terminates batch subsystems associated with the IRLM. Batch database backout is required for only those batch executions that perform database updates and do not use dynamic backout.

After you restart the failed IRLM, enter an MVS `MODIFY` command, specifying the `RECONNECT` keyword, for each IMS online system. This command reestablishes the data-sharing environment, and initiates reprocessing for the suspend queue.

Restart in a Sysplex Data-Sharing Environment

In a sysplex data-sharing environment, an IRLM is informed in any of the following situations:

- MVS failure
- IMS failure
- Failure of another IRLM
- Loss of connection to physical locking resources

The IRLM affected by the failure disconnects from the data-sharing group. The remaining IRLMs in the data-sharing group are referred to as surviving IRLMs.

When an IRLM is forced to leave a data-sharing group for any reason, that IRLM issues the `DXR136I` message, which says that an IRLM has disconnected from the data-sharing group.

The surviving IRLMs issue the `DXR137I` message, which says the group status has changed, and that an IRLM has been disconnected from the data-sharing group.

No operator action is required for either message. As in a nonsysplex data-sharing environment, IRLM protects resources that are locked from being accessed by other IMS subsystems.

Normal restart procedures can be used to restart the IRLM and its associated IMS.

Recovering from Sysplex Data-Sharing Failures

The following types of failures can occur in a sysplex data-sharing environment:

- Coupling facility connection failure
- Coupling facility structure failure
- Failure in rebuilding a coupling facility
- Coupling facility connection failure during IMS restart
- Failure when IMS tries to identify to the IRLM
- MVS or other failure requiring system recovery

If multiple failures occur simultaneously (for example, IRLM and OSAM structures fail on the same coupling facility), each component (IRLM, IMS, and so on) takes the appropriate recovery action.

Coupling Facility Connection Failures

Various types of problems can cause the connection between IMS and the coupling facility to be lost. For example, the connection is lost if the physical link (a fiber optic cable) is cut, after power failures, hardware check stops, coupling facility microcode failures, and similar events.

If there are multiple links between an IMS and the coupling facility, loss of one link has no effect on IMS or the IRLM. However, if the link that is lost is the **last** link, it must be recovered before data sharing on that IMS can be restarted. Processing still continues for all other IMS subsystems in the data-sharing group.

When the last connection between IMS and the coupling facility is lost, what happens depends on the IMS operating environment:

Batch Environment
All IMS batch jobs that were connected to the coupling facility terminate with a `U3303` abend.

DB/DC or DBCTL Environment
If the connection was between an IMS and an OSAM or VSAM structure, IMS drops out of data sharing. Any transactions trying to access shared data terminate with a `U3303` abend. However, transactions accessing nonshared databases can still be processed.

If the connection that failed was between an IRLM and the IRLM lock structure, the IRLM disconnects from the data-sharing group. Any transactions trying to access shared data terminate with a `U3303` abend.

When connection between the coupling facility and IMS is reestablished, MVS notifies IMS and IMS resumes data sharing.

Example: Figure 16 shows the connection loss between IMS, OSAM, and VSAM structures as a result of a coupling facility connection failure.

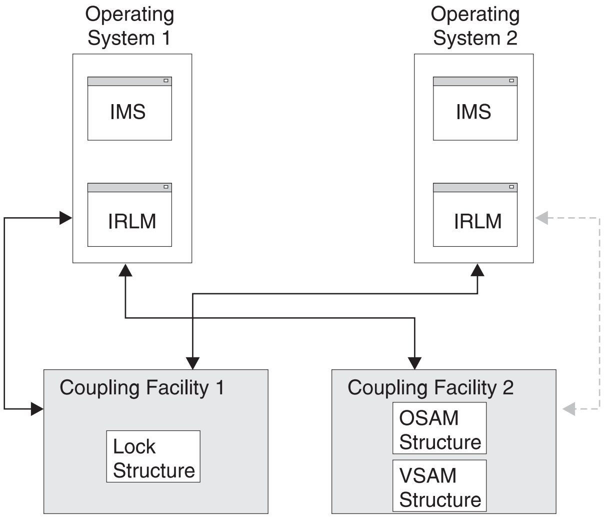


Figure 16. Connection Loss between IMS and OSAM and VSAM Structures

The connection that is lost is the link between MVS2 and coupling facility CF02. When the last link fails:

- IMS2 drops out of data sharing and only nonshared databases can be accessed on the MVS2 system. IMS1 and IMS3 continue data sharing.
- Transactions trying to access shared data on MVS2 terminate with a `U3303` abend.

In this circumstance, you receive the `DATA SHARING STOPPED` message. You can enter the MVS `DISPLAY XCF,STRUCTURE,STRNAME=` command to determine what failed. The displayed output, as shown in Figure 17 on page 233, shows that the MVS2 link to the coupling facility failed. MVS1 and MVS3 links are still active.

```
11:44:37.84 D XCF,STRUCTURE,STRNAME=OSAMSESXI

   ⋮

             & AMPERSAND DENOTES CONNECTOR WHO LOST CONNECTIVITY TO STRUCTURE
             CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
             ---------------- -- -------- -------- -------- ------------
             IMS1             03 0003000D MVS1     DLIOCSA8 0066 ACTIVE
            &IMS2             01 0001001D MVS2     DLIOCSB8 0066 FAILING
             IMS3             02 00020014 MVS3     DLIOCSC8 0066 ACTIVE
```

Figure 17. Output When Displaying a Coupling Facility Connection Failure

In the online environment, the MVS2 link to the OSAM structure is reestablished, MVS notifies IMS, and data sharing resumes. In the batch environment, you need to resubmit the batch job when the link is reestablished. You will receive the `CF INITIALIZATION COMPLETE` message, indicating IMS has resumed data sharing. If you issue a `DISPLAY XCF,STRUCTURE` command, display output shows that MVS2 is connected (ACTIVE).

Coupling Facility Structure Failures

When a structure fails, what happens depends on the operating environment:

Batch Environment
All IMS batch jobs that were connected to the coupling facility terminate with a `U3303` abend.

DB/DC or DBCTL Environment
All IMSs in the data-sharing group temporarily suspend data sharing. If the structure that failed was an OSAM or VSAM structure, IMS tries to rebuild the structure. If the rebuild is successful, all IMSs connect to the new structure and data sharing continues. If the structure that failed was an IRLM structure, all IRLMs try to rebuild the IRLM structure. If the rebuild is successful, all IRLMs connect to the new structure and locking for data sharing continues.

Example: Figure 16 on page 232 shows the configuration for an OSAM structure failure. When the OSAM structure on coupling facility CF02 fails, the following happens:

- All IMSs suspend data sharing. If you display the status of the structure at this point (using the `DISPLAY XCF,STRUCTURE,STRNAME=OSAMSESXI` command), output shows that the structure is in the quiesce phase of rebuild, as shown in Figure 18 on page 234.

```
08:50:59.29 D XCF,STRUCTURE,STRNAME=OSAMSESXI
08:51:13.32 IXC360I  08.50.59  DISPLAY XCF 143
            STRNAME: OSAMSESXI
             STATUS: REASON SPECIFIED WITH REBUILD START:
                       STRUCTURE FAILURE
                     REBUILD PHASE: QUIESCE
              POLICY SIZE    : 2048 K
              PREFERENCE LIST: CF02     CF01
              EXCLUSION LIST IS EMPTY


   ⋮

             * ASTERISK DENOTES CONNECTOR WITH OUTSTANDING REBUILD RESPONSE
             CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
             ---------------- -- -------- -------- -------- ---- -------
            *IMS1             03 00030015 MVS1     DLIOCSA8 003A ACTIVE
            *IMS2             01 00010026 MVS2     DLIOCSB8 0038 ACTIVE
            *IMS3             02 0002001D MVS3     DLIOCSC8 0038 ACTIVE
```

Figure 18. Output When Displaying a Structure Failure

- When all IMSs suspend data sharing, a new structure is built. All IMSs are then connected to the new structure and data sharing resumes. If you display the status of the new structure after rebuilding, output is similar to that shown in Figure 19.

```
08:52:39.78 D XCF,STRUCTURE,STRNAME=OSAMSESXI
08:52:40.43 IXC360I  08.52.39  DISPLAY XCF 168
            STRNAME: OSAMSESXI
             STATUS: ALLOCATED
              POLICY SIZE    : 2048 K
              PREFERENCE LIST: CF02     CF01
              EXCLUSION LIST IS EMPTY

             ACTIVE STRUCTURE
             ----------------
              ALLOCATION TIME: 03/25/94 08:51:27
              CFNAME         : CF01
              COUPLING FACILITY: ND01...
                                 PARTITION: 0    CPCID: 00
              ACTUAL SIZE    : 2048 K


   ⋮

             CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
             ---------------- -- -------- -------- -------- ------------
             IMS1             03 00030015 MVS1     DLIOCSA8 003A ACTIVE
             IMS2             01 00010026 MVS2     DLIOCSB8 0038 ACTIVE
             IMS3             02 0002001D MVS3     DLIOCSC8 0038 ACTIVE
```

Figure 19. Output When Displaying a New Structure after Rebuilding

Failure in Rebuilding a Coupling Facility

You can initiate a rebuild of the coupling facility by issuing the MVS `SETXCF START,REBUILD` command. MVS can also initiate a rebuild in response to:

- A coupling facility structure failure
- A connection failure if the active System Failure Management (SFM) policy threshold for rebuilding the coupling facility has been reached

If the rebuild fails, IMS stops data sharing. Transactions trying to access shared data terminate with a U3303 abend. If the rebuild failure was associated with an OSAM or VSAM structure, transactions accessing nonshared databases can still be processed. If the failure was associated with an IRLM structure, the IRLM disconnects from the data-sharing group.

Coupling Facility Connection Failure during IMS Restart

If the connection of the active system to the coupling facility fails during IMS restart, restart processing continues. If the failure is on the XRF alternate subsystem and the alternate subsystem is unable to connect to the coupling facility, restart cannot proceed.

Failure When IMS Tries to Identify to an IRLM

A failure occurs if IMS cannot identify itself to the IRLM. When this happens, IMS issues a message indicating the reason for the failure and then abends. You can gather information to determine what action to take using the DISPLAY XCF,STRUCTURE and MODIFY irlmproc*x*,STATUS commands.

Except for this identify failure, IRLM failure processing is the same as with nonsysplex data sharing. IMS:

- Backs out all uncommitted database changes.
- Removes authorization of all shared databases through DBRC. The system is stopped.
- Waits for the IRLM to be restarted.

MVS and Other Failures Requiring System Recovery

If MVS fails or other failures occur that require system recovery (for example, a power outage), IMS processing is the same as it is in the nonsysplex data-sharing environment. For the failing system, operations consist of:

- Restarting MVS
- Restarting the IRLM
- In the batch environment, backing out batch jobs if necessary
- In the IMS DB/DC or DBCTL environment, emergency restarting IMS (using the /ERESTART command)

You can restart MVS and the IRLM on a different system. Also, batch backout and emergency restart can be done on any MVS and IRLM in the data-sharing group.

For the non-failing systems:

- Locks are retained for the failed IMS.
- Locks are released as the failed IMS restarts and recovers its databases.
- Database buffers for the failed IMS are invalidated.
- Database error conditions are sent to the other IRLMs.

Chapter 15. Extended Recovery Facility (XRF)

To operate IMS in an XRF complex, you must understand what tasks to perform during the six XRF phases:

Initialization:

- How to bring up the active IMS subsystem
- How and when to bring up the alternate IMS subsystem

Synchronization:

- Whether or not to enter the `/CHECKPOINT SNAPQ` command

Tracking:

- How to use the `/DISPLAY` command and read the status line of the master terminal
- How to use online change
- How to change surveillance dynamically
- How to use other XRF-related commands
- How to use XRF in an IMS DBCTL environment
- How to use XRF in a data-sharing environment

Takeover:

- How does automatic takeover differ from manual takeover
- How to work with the operator of the other subsystem and the operator of the network
- How to initiate a takeover
- How to ensure I/O prevention on the failed active subsystem
- When to end I/O toleration on the alternate subsystem
- How to switch DASD and class-3 terminals

Post-takeover:

- How to ease the workload on the new active subsystem
- How and when to restore service to class-3 and class-2 terminals that failed to switch
- How to complete dynamic DL/I backouts
- How to perform diagnostics on the failed subsystem
- How to bring up a third subsystem as an alternate or to restore the original active as the alternate subsystem

Termination:

- How to terminate an XRF complex
- How to stop surveillance

XRF requires the following resources:

- MVS/ESA, IMS, and VTAM, running on each central processor complex (CPC).
- If USERVAR is used, Network Control Program (NCP) running on a 3725 Communication Controller for full terminal switching.
- IMS system logs— online data sets (OLDSs) and write-ahead data sets (WADSs)—and certain other system data sets must be on DASD shared by both IMS subsystems.

Related Reading: This chapter includes information on the education and procedures that you need to run an XRF complex. For information on these topics from a systems level, see the XRF information in *IMS Version 9: Administration Guide: System*.

Planning for Operating an XRF Complex

There are several considerations when you plan changes to your operational control for XRF. These considerations are:

- Whether additional staff should be responsible for an alternate subsystem and assume full control when that subsystem becomes the active
- Whether a relocation of the IMS master terminal and MVS alternate subsystem console is necessary
- How additional takeover activities and other procedure changes are to be incorporated into "run books"
- Whether a takeover should proceed automatically or by means of operator intervention when the alternate subsystem makes a takeover decision

When operating with XRF, it is important that the staff understand the "single-system image" and have an awareness of what physically belongs to active or alternate subsystems, what is shared, and generally where system and database data sets are located. The operator preparation is mostly knowing the division of responsibility when processing moves to an a different processor as a result of a takeover.

Regardless of whether master terminal operations staff have master terminals close to one another, close communication is required.

Establishing Communication Procedures among Operators

To promote good communication among the operators of an XRF complex, place all consoles in a central location. The speed with which the takeover proceeds depends on operators confirming that a takeover is in progress and that I/O prevention is complete. This confirmation is easiest when operators are located in the same physical area.

Figure 20 on page 239 shows a recommended configuration for the consoles that are responsible for managing XRF:

- A group of four logical consoles:
  - IMS master terminal for the active subsystem
  - IMS master terminal for the alternate subsystem
  - Multiple console support (MCS) console that receives messages from MVS for the active subsystem
  - Multiple console support (MCS) console that receives messages from MVS for the alternate subsystem
- IMS secondary master terminal with attached printer for the active subsystem
- IMS secondary master terminal with attached printer for the alternate subsystem
- Network terminal for VTAM and the Network Communication Control Facility (NCCF)
- Optionally, two additional consoles for JES3 in the two systems (not shown in the figure)

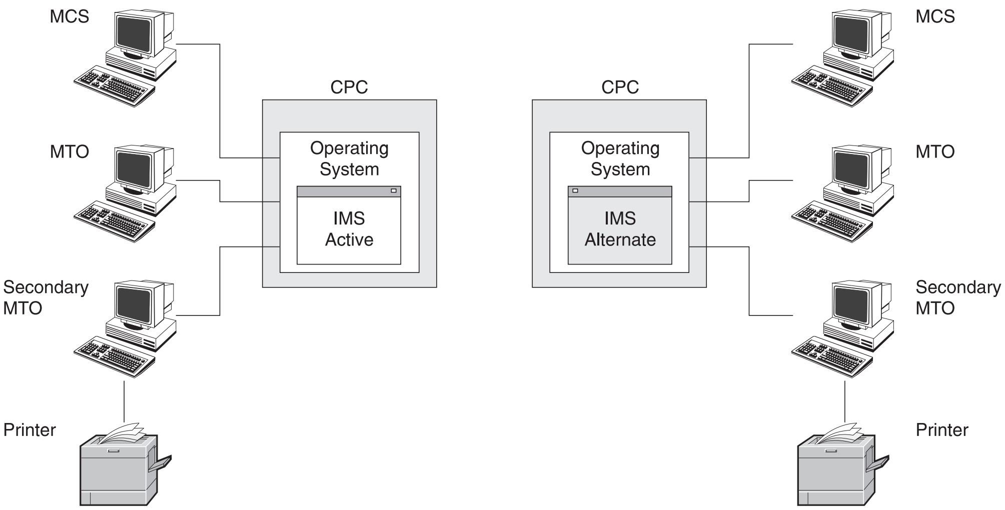


Figure 20. Recommended Console Configuration for XRF

It is important that the active and alternate IMS subsystems each have their own IMS master terminals. IMS limits the commands that the MTO can issue on the alternate subsystem.

On this set of terminals, your operators have a collective view of MVS, IMS, and network events. Your operators cannot issue the MVS `SYSTEM RESET` command on the processor consoles, but they can receive all the availability manager (AVM) messages and can issue monitoring commands.

To understand how important it is to have the operators' consoles together, consider the implications of having separate centers for the active subsystem and the alternate subsystem operations. Operators at each center must have some understanding of the entire XRF complex. The state of the alternate subsystem is important to the operators for the active subsystem, and vice versa. For example:

- The operators for the active subsystem should know when the initialization of the alternate subsystem is complete.
- The operators for the active subsystem must know of a failure of the alternate subsystem.
- The operators for the alternate subsystem must know of the completion of I/O prevention in the failing active subsystem.

With the operators in two separate physical locations, the chance for misunderstanding or lack of communication is great. The success of the takeover and the integrity of the databases lies with your operators and their awareness of what is going on in the entire complex.

At changes of shifts, the operators on one shift must let the operators on the following shift know the current state of the complex.

Normal IMS Operations in an XRF Complex

In terms of command usage, the operator needs to know the following:

- When and how to initiate a takeover manually
- When and how to terminate an IMS alternate subsystem
- What terminal switching needs to be done
- What database control might be required

- How to interpret /DISPLAY output when operating in an XRF complex
- How to make changes to the takeover criteria and surveillance methods
- How to respond to a failure of the takeover process

During takeover, operators must be aware of response time, availability of terminals, data set access problems, and database access. You can bring up the failed active subsystem as a new active subsystem in parallel with the alternate subsystem, providing interim end-user service.

Those terminals considered critical for continuous service should be defined as class-1 terminals. However, a certain number of these terminals might have to be switched manually. A convenient way of connecting these terminals to the new active IMS is to use an automated operator program. This program could issue a set of IMS /START or UPDATE commands. Another use of automated operator programs could be to modify surveillance intervals and takeover criteria. See Chapter 13, “Automated Operations,” on page 185.

Making Online Changes in an XRF Complex

In addition to preparing the staging libraries, you need to prepare the inactive libraries.

Using XRF in a Data-Sharing Environment

If you are using IRLM 2.1 in a sysplex data-sharing environment and the alternate subsystem is available, there are no required tasks to maintain continued data availability.

Using XRF with a Data Propagation Manager

To propagate changed data from IMS DL/I databases in an XRF complex to DB2 tables, be sure IMS and DB2 are available to both the active and alternate subsystems. If DB2 is not available on one of the systems, the data propagation from IMS to DB2 fails until DB2 is made available.

Using XRF in an IMSplex

You can use XRF in an IMSplex. XRF can be used in an IMSplex in the same way that it is used on a local system. The XRF alternate needs to be defined in the IMSplex and have access to the Common Service Layer (CSL). A Structured Call Interface (SCI) must reside on the MVS image where the XRF alternate resides. For more information on configuring an IMSplex, see *IMS Version 9: Common Service Layer Guide and Reference*.

An XRF system registers for IMS commands with the Operations Manager (OM) using its IMSID. When commands are routed to an XRF active or alternate system, the IMSID should be used on the ROUTE().

For owned resources, an XRF system defines its resources to the Resource Manager (RM) using the RSENAME as the resource owner. When the XRF alternate system takes over, it takes over ownership of the resources.

You cannot use XRF to recover an entire IMSplex, but you can use XRF to recover an individual IMS in an IMSplex.

HALDB OLR cannot be initiated on an XRF alternate system. For more information about HALDB OLR support, see *IMS Version 9: HALDB Online Reorganization Guide and Reference*.

Initializing XRF

During the initialization phase, in addition to standard IMS initialization tasks, the active subsystem and the alternate subsystems in an XRF complex perform tasks specifically related to XRF. You need to know about the following:

- What IMS parameters are in effect for XRF
- How to initialize MODSTAT, RDS, and the message-queue data sets
- Whether or not IMS joins the availability manager
- How to successfully start the XRF complex
- Initializing the USERVAR or Specifying Multi-node Persistent Sessions (MNPS)

Determining XRF Parameters

The active subsystem uses the recoverable service element name (RSE name), not the MVS SSID, when signing on to DBRC and identifying to IRLM. Using the RSE name allows the alternate subsystem to inherit databases and their authorizations from the active subsystem during takeover.

Both subsystems read IMS.PROCLIB member DFSHSBxx for their RSEname, surveillance options, and takeover conditions.

Related Reading: For a description of the DFSHSBxx member, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

Initializing Data Sets

During initialization, the active subsystem is startable but not XRF-capable if it does not find two MODSTAT data sets (unless global online change is enabled, in which case the MODSTAT data sets are not needed), two restart data sets (RDSs), and, if you use IMS Fast Path, four MSDBCPn data sets. The alternate subsystem, however, is not startable unless it finds these data sets on shared DASD.

Both the active subsystem and the alternate subsystem allocate latest MODSTAT and RDS, but the alternate subsystem switches to the unused pair during startup. The active subsystem uses the last active MSDBCPn pair, and the alternate subsystem uses the other two.

In a shared-queues environment, the active IMS subsystem does not need the message-queue data sets, but the alternate subsystem IMS subsystem does use these data sets because the alternate subsystem does not participate in sharing the IMS message queues until it takes over the active IMS subsystem workload.

Joining the Availability Manager

The availability manager (AVM) has its own address space. You can start the AVM with the MVS `START` command or let MVS start it automatically when you issue the IMS `/NRESTART` or `/ERESTART` command. To start the availability manager, issue the MVS `START AVM` command. When the AVM is initialized, you receive the `AVM001I` message.

Recommendation: Use the MVS START AVM,SUB=MSTR command to start the AVM. If you omit the SUB=MSTR keywords, MVS starts the AVM from the JES address space, and the resulting dependency on JES means you cannot stop JES without also stopping the AVM.

Both the active subsystem and the alternate subsystem notify the MVS availability manager of their participation in an RSE during initialization. When IMS successfully joins the AVM and RSE, you receive messages DFS3871I and DFS3873I.

If IMS fails to connect to the AVM, you receive message DFS3872I. If the connection fails, the operator must initiate I/O prevention during XRF takeover. You must have a procedure established in case the connection fails during initialization of either the active subsystem or the alternate subsystem. You might want to determine the cause of the failure and restart the initialization to ensure that the connection is successful.

Starting the Subsystems in an XRF Complex

Start the active subsystem first. Use the /ERE BACKUP command to start the alternate subsystem only after the active subsystem startup completes with the DFS994I message.

Recommendation: If the active subsystem begins processing before you start the alternate subsystem, issue the /CHECKPOINT command on the active subsystem to take a simple checkpoint. If you establish a checkpoint before starting the alternate subsystem, the alternate subsystem needs to read only the active subsystem's log from this checkpoint.

After an XRF takeover, restart the alternate subsystem first if it issued a simple checkpoint, even if you had not yet released it from I/O toleration (by issuing the /UNLOCK SYSTEM command or replying to an AVM message). Otherwise, restart the active subsystem first.

When you start an XRF alternate subsystem (using the /ERE BACKUP command), IMS establishes VTAM backup sessions (if USERVAR is used with XRF), an AVM node, and an ISC link to facilitate communications with the active subsystem. If the alternate subsystem fails and is restarted by the OS/390 Automatic Resource Manager (ARM), the VTAM ACB and the VTAM master terminals will be opened towards the end of the alternate subsystem's initialization.

Startup Procedures for Subsystems

Because an RSE in an XRF complex consists of two IMS subsystems, and because each must perform different activities, you must have two distinct procedures to start them up:

- One of these procedures must specify HSBID=1 in the IMS procedure and, in a non-shared-queues environment, contain DD statements for its set of unique message-queue data sets (LGMSG, SHMSG, QBLKS, LGMSGL, SHMSGL, and QBLKSL).
- The other procedure must specify HSBID=2 in the IMS procedure and contain DD statements for its set of message-queue data sets, which must be distinct from the data sets listed above.

In an XRF complex, the OLDS definition (OLDSDEF) for the active subsystem must match the OLDSDEF for the alternate. Also, all OLDSs used by the active subsystem must be accessible to the alternate subsystem, regardless of whether they are in the OLDS definition.

The IMS Identifier

The IMS ID identifies a particular IMS subsystem. Specify the subsystem name in the IMSCTRL macro during system generation, and you can overwrite the subsystem name in JCL procedures. When an IMS BMP job tries to link to an IMS subsystem, it attempts to link first to the subsystem specified by the `IMSID=` parameter, and next to any subsystem specified by the `ALTID=` parameter.

If the active and alternate subsystems in an XRF complex are on the same central processor complex (CPC), their IMSIDs or ALTIDs must be different. If the subsystems are on different CPCs, you can specify either the same or different IMSIDs or ALTIDs.

If you specify different IMSIDs or ALTIDs, you must specify the correct IMSID or ALTID in all dependent region JCL procedures.

Related Reading: For more information about the IMSID and ALTID, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

Managing Terminal Sessions for XRF

You can use VTAM USERVAR or VTAM Multi-node Persistent Sessions (MNPS) to manage terminal sessions for XRF. In an XRF system with USERVAR, the alternate subsystem continuously maintains an open backup session for class-1 terminals and switches the terminals to that session during a takeover. In an XRF system with MNPS, the alternate subsystem does not maintain backup sessions. Instead, the alternate subsystem opens a new instance of the MNPS ACB during a takeover. Sessions for class-1 terminals are then rerouted through the new MNPS ACB on the alternate IMS. The differences between each method are transparent to the end user.

Initializing the USERVAR to Manage Terminal Sessions for XRF

VTAM uses a variable called the USERVAR to associate a user’s logon message with the application name of the currently active IMS subsystem. The two subsystems in an XRF complex have different application names.

During IMS initialization (and later, during XRF takeover), to allow terminals to log on to the correct subsystem, IMS or network operators must initialize the VTAM USERVAR tables. If you are not using the VTAM USERVAR Management Enhancement in your network, the network operator must enter the `MODIFY` *`procname`*`,USERVAR` command for the alternate subsystem VTAM if it owns terminals. You might also have to tell operators for other VTAMs that communicate with the XRF complex to enter the MVS `MODIFY` command to change the application name in their USERVAR tables.

The active IMS subsystem issues a `MODIFY` command to tell its own VTAM to initialize the application name in its USERVAR table.

If you are using the VTAM USERVAR Management Enhancement in your network, the network operator should not initialize the VTAM USERVAR table nor use an NCCF program (CLIST) to propagate the table to other VTAM nodes. It is also not necessary for the operator to enter the MVS `MODIFY` *`procname`*`,USERVAR` command to change the application name in the USERVAR tables. The VTAM USERVAR Management Enhancement does all of this for IMS.

| ### Specifying MNPS to Manage Terminal Sessions for XRF

| To have IMS use MNPS for XRF terminal switching, specify the `MNPS=`*`name`* optional
| keyword on the DFSPBxxx or DFSHSBxx IMS.PROCLIB members. The *name*

specifies the name of the MNPS ACB, which is used for terminal recovery. The MNPS ACB name must be the same for the active IMS and the alternate IMS, and only one MNPS ACB can exist at a time. If the `MNPS=` keyword is specified, the `USERVAR=` keyword is ignored.

Actions Performed during Synchronization

During synchronization, the operator might need to perform the following actions:

- Take a checkpoint (`/CHECKPOINT SNAPQ`).
- Respond to inconsistent system definitions.
- Respond to improperly allocated system data sets.

The alternate subsystem system automatically performs the following actions:

- Preopening backup sessions for class-1 terminals, if using USERVAR
- Pre-allocating and preopening DBDSs

Taking a Checkpoint (/CHECKPOINT SNAPQ)

After you start the alternate subsystem with the `/ERESTART BACKUP` command:

- If you do not have an intersystem communication (ISC) link, and you receive message `DFS3801I` on the alternate subsystem (saying that a checkpoint SNAPQ is required on the active subsystem):
  1. Switch OLDSs to reduce contention on the OLDSs.
  2. Issue the `/CHECKPOINT SNAPQ` command on the active subsystem.
- If you do have an ISC link between the active subsystem and the alternate subsystems, and if the most recent checkpoint on the OLDS is not a SNAPQ, then the alternate subsystem sends the active subsystem a synchronization request to take a SNAPQ checkpoint.

  If the alternate subsystem finds a SNAPQ (no matter how old it is) before finding any other checkpoint, it applies changes from the log beginning with that SNAPQ. Because this processing might stress your system, be careful when you start the alternate; you might not want to start it during peak processing hours.

  If the most recent checkpoint is a SNAPQ, you can reduce processing during initialization by taking a simple checkpoint (using the `/CHECKPOINT` command) on the active subsystem before starting the alternate subsystem.

The active subsystem sends an “OKTOSYNC” message to the alternate subsystem over the ISC link when checkpoint processing is in progress. If you do not have an ISC link, the alternate subsystem monitors the RDS to determine when the checkpoint is taken.

In a shared-queues environment, the `/CHECKPOINT SNAPQ` command synchronizes the active subsystem and alternate subsystems, but does not copy the message queues to the IMS log.

Checking System Definitions

The alternate subsystem uses the information on the OLDS from the most recent SNAPQ to ensure consistency of system definition for each IMS subsystem. If the definitions are inconsistent, the `/ERESTART BACKUP` command fails. System definition for an XRF complex includes the following keywords, parameters, and statements:

- Table 36 shows the XRF definition in the system definition macros:

Table 36. Keywords, Parameters, and Statements for an XRF Complex in the System Definition Macros

Required	Optional
HSB= (IMSCTRL macro)	BACKUP=(CTLUNIT, LINEGRP, MSLINK, MSPLINK, STATION, and TYPE macros)
APPLID= and PASSWORD= (COMM macro, if you use VTAM)	BACKUP= and ADDR= (LINE macro)
	BACKUP=, ADDR=, and NAME= (TERMINAL macro)
	BACKUP= (ETO logon descriptor)

- Table 37 shows the XRF definition in the IMS.PROCLIB member DFSHSBxx:

Table 37. Keywords, Parameters, and Statements for an XRF Complex in the IMS.PROCLIB member

Required	Optional
RSENAME=. Specifies the RSE name, which must be the same for both the active and alternate subsystems.	ALARM=. Specifies whether the service processor alarm sounds for a takeover request.
	AUTO=. Specifies whether a takeover requires your approval.
	DEFERFIX=. Identifies the IMS.PROCLIB member DFSFIXnn.
	USERVAR=. Specifies the USERVAR (if you use VTAM)
	SURV=. Specifies the surveillance mechanisms.
	LNK=. Specifies interval and timeout values for links.
	LOG=. Specifies interval and timeout values for the log.
	RDS=. Specifies interval and timeout values for the RDS.
	SWITCH=. Controls whether absences of surveillance signals or failures of VTAM or of the IRLM cause the alternate subsystem to request or initiate a takeover.

  The IMS.PROCLIB member DFSHSBxx controls the following aspects of the XRF complex:

  – Surveillance mechanisms

  – Takeover conditions

  – Display of optional messages

  – VTAM USERVAR name or MNPS ACB name (depending on whether USERVAR or MNPS is used)

  – Name of the IMS.PROCLIB member that determines non-Fast Path control region page-fixing values for the alternate subsystem after a takeover

  – Name by which the complex is known to MVS, DBRC, and the IRLMs

- In the IMS.PROCLIB member DFSFIXnn:

This member specifies page-fixing options for the alternate subsystem after a takeover. You can move non-Fast Path related options of an existing DFSFIXnn member to this additional page-fix member to reduce the real storage requirements of the alternate subsystem during initialization and tracking. Use this member to fix buffer pool pages that you do not want fixed during initialization and tracking.

- In the IMS.PROCLIB member DFSVSMxx:

  If the active and the alternate subsystems are on the same CPC, you must change the `VSAMPLS` keyword on the `OPTIONS` control statement to specify `LOCL` (local) instead of `GLBL` (global) for at least one of the subsystems. Only one control region on a processor can request a VSAM Shared Resource Pool. If you specify `LSO=Y` or `LSO=S` for the IMS control region, the default for `VSAMPLS` is `LOCL`.

  The `OLDSDEF` and `WADSDEF` control statements must be identical.
- In the `EXEC` statement for each control region:
  - `HSBMBR=`, which gives the suffix for the DFSHSBxx member of IMS.PROCLIB.
  - `HSBID=`, which identifies the message queues. Setting `HSBID=` to null deactivates XRF capability.

    If you do not set `HSBID=` to null, you must specify either `1` or `2`. The procedure for IMSA must specify a different value than you specify for IMSB.

    If after a takeover, you bring up a third subsystem as an alternate subsystem, specify the same `HSBID=` as on the failing active subsystem.
- If the XRF subsystems participate in block-level data sharing, you need JCL statements for the IRLMs. See *IMS Version 9: Administration Guide: System*.
- DD statements for system data sets. For data set placement requirements, see *IMS Version 9: Administration Guide: System*.

Related Reading: For complete descriptions of XRF system definition parameters and statements, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

Checking System Data Sets

If the RDS, MODSTAT, and MSDBCPn data sets are not properly allocated on the active and alternate subsystems, the alternate subsystem terminates.

The alternate subsystem loads MSDBs from the SNAPQ records rather than from the MSDBCPn data set containing the last image.

Because IMS allocates data sets and pre-loads programs during initialization of dependent regions, you should exercise care if you make any changes in these areas after this time. Specifically:

- If programs running on the active subsystem modify data set allocation information, you might have to rewrite these programs to use dynamic allocation. Also note that IMS does not perform I/O prevention for any user data set.
- If you rebind preloaded programs on the active subsystem and subsequently activate them using the MVS `START` command, you must also restart the associated regions on the alternate subsystem. The alternate subsystem does not automatically start or stop its dependent regions based on actions on the active subsystem.

Preopening Backup Sessions for Class-1 Terminals

If USERVAR is used, the alternate subsystem establishes backup sessions for terminals that have the `BACKUP=(, YES)` option specified and that have open sessions (as recorded in the SNAPQ log records).

Tracking the XRF Complex

During the tracking phase, your tasks are to monitor, maintain, and modify the XRF complex. Because these tasks involve IMS commands that are also useful during other processing phases, this section reviews all XRF-related IMS commands.

During the tracking phase, operators can issue the following commands to monitor the system:

- IMS `/DISPLAY HSB` command
- VTAM `DISPLAY NET,ID` and `DISPLAY NET,USERVAR` commands (if USERVAR is used)

These commands display the status of the XRF complex and monitor the status of sessions on class-1 terminals.

During normal operations, the operators do not have any tasks specifically for XRF, nor do they receive messages about XRF.

Operator Commands on an Alternate Subsystem

Because the alternate subsystem does not process transactions, the operator for the alternate subsystem does not need some of the commands that the operator for the active subsystem needs. In fact, IMS prevents the operator for the alternate subsystem from entering certain commands, displaying the `DFS3824I` message that says that the command is not valid for the backup system.

You can enter the following IMS commands on an alternate subsystem:

```
/ACTIVATE
/CANCEL
/CHNAGE
/CLSDST
/COMPT
/DEQUEUE
/DISPLAY
/END
/ERESTART
/FORMAT
/IDLE
/NRESTART
/OPNDST
/PSTOP
QUERY AREA
QUERY DB
QUERY TRAN
/RCLSDST
/RCOMPT
/RDISPLAY
/RMLIST
/RSTART
/SMCOPY
/TEST
/TRACE
UPDATE
```

You can enter the following IMS commands on an alternate subsystem for VTAM master and secondary master terminals:

```
/RSTART NODE x
/START NODE x
/STOP NODE x
```

You can enter the following IMS commands on an alternate subsystem for BTAM master and secondary master terminals:

```
/RSTART LINE x PTERM y
/START LINE x PTERM y
```

```
/STOP LINE x PTERM y
```

Monitoring an XRF Complex

Monitoring an XRF complex requires that you know how to read the status line of the master terminal, and how to use the /DISPLAY command.

Reading the System Status Line

The top line of the screen of the IMS master terminal in an XRF complex gives the following information:

- The current date
- The current time
- The RSE name of the active and alternate subsystems
- The status of the subsystem:
  - ACTIVE for the active subsystem
  - BACKUP for the alternate subsystem
- The phase of the active and alternate subsystems
  - Before restart:

    INITIALIZATION
  - After starting the active subsystem:

    Null
  - After starting the alternate subsystem:

    AWAITING SNAPQ

    SYNCHRONIZATION

    TRACKING IN PROGRESS

    TAKEOVER REQUESTED

    TAKEOVER IN PROGRESS
  - After XRF takeover:

    AWAITING I/O PREVENTION (on the active subsystem)

    I/O TOLERATION (on the alternate subsystem)
- The IMS subsystem identifier, such as IMSA

Related Reading: For a complete description of the system status line, see *IMS Version 9: Command Reference*.

Displaying Tracking and Other XRF Complex Information

You can use the /DISPLAY command to display relevant tracking and other information for an XRF complex.

/DISPLAY STATUS DATABASE
/DISPLAY DATABASE dbname BKERR

Use either one of these commands to display information about the following:

- Incomplete backouts that prevent use of a database
- EEQEs for a database

To complete backout, restart any database displayed. If the /START DB or UPDATE DB START(ACCESS) command to start the database fails because the required OLDS have been archived and reused, use the date and time shown in the display to determine which SLDS to use and then perform batch backout.

/DISPLAY ACTIVE

Use this command on either the active or the alternate subsystem in an XRF complex to display information about active regions and, after the time stamp, the RSE name, the status of the subsystem (ACTIVE or BACKUP), and the phase of the XRF complex. The status can be one of the following:

AWAITING SNAPQ
SYNCHRONIZATION
TRACKING
TAKEOVER REQUESTED
TAKEOVER IN PROGRESS
I/O TOLERATION

/DISPLAY HSB

Use this command on either the active or the alternate subsystem to display:

- The RSE name
- The status of the subsystem (ACTIVE or BACKUP)
- The mode (whether the alternate subsystem is in I/O toleration after a takeover)
- The processing phase of the alternate subsystem (if entered on the alternate subsystem), which is one of the following:

  AWAITING SNAPQ
  SYNCHRONIZATION
  TRACKING IN PROGRESS
  TAKEOVER REQUESTED
  TAKEOVER IN PROGRESS
- The IMSIDs of the active subsystem and the subsystem used to enter the command
- The VTAM USERVAR, if USERVAR is used
- The MNPS ACB name, if MNPS is used
- The time associated with the log record currently being read by the alternate system as it tracks the active subsystem
- The surveillance mechanisms in effect for the complex
- The takeover conditions specified on the `SWITCH=` keyword of the DFSHSBxx member of IMS.PROCLIB

Related Reading: For a detailed description of these commands and the information they display, see *IMS Version 9: Command Reference*.

Monitoring the Status of the Network

Some VTAM messages have a STATUS field that appears in response to the VTAM `DISPLAY` command. Examples in the following sections describe the XRF-related keywords for the `DISPLAY` command and the STATUS fields that appear in the resulting messages. The examples below refer to the sample XRF complex shown in Figure 21 on page 250. The complex contains a primary and a backup session; IMSNAME1 is in the primary session and IMSNAME2 is in the backup session.

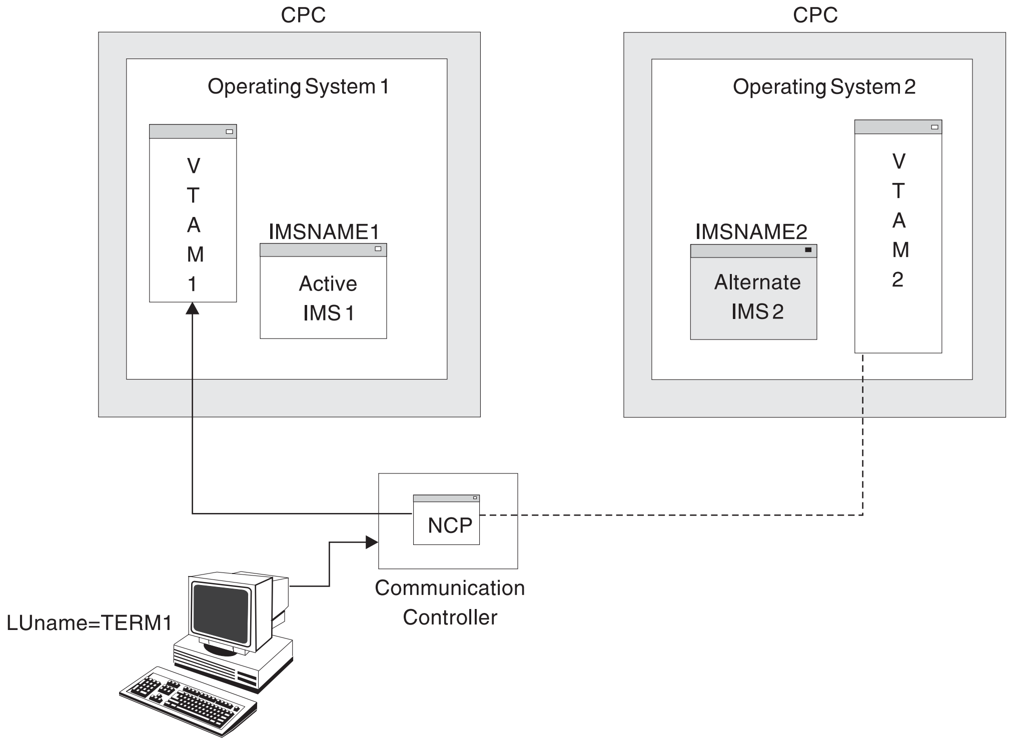


Figure 21. Monitoring Status

Displaying the Status of Specific Terminals

The VTAM `DISPLAY NET,ID` command allows the network operator to see the status of the primary and backup sessions for a specific class-1 terminal. For example, to display the status of both sessions for the class-1 terminal with logical unit name TERM1, the network operator issues the following command:

```
DISPLAY NET,ID=TERM1,SCOPE=ALL
```

The response to this command is the following message:

```
IST634I NAME       STATUS   SESSION ID SEND  RECV  VRN  TP  . . .
IST635I IMSNAME1  APRIM-PRI
IST635I IMSNAME2  ABKUP-PRI
IST8351 END
```

Except for the STATUS field, the two `IST635I` messages should be familiar to your network operators. The NAME and STATUS fields in the two messages indicate the following:

- The following is the first IST635I message:

  ```
  IST635I  IMSNAME1 APRIM-PRI
  ```

  In this message:

  – `IMSNAME1` indicates that terminal TERM1 is in session with IMSNAME1.

  – `A` indicates that the session is active, that is, not pending.

  – `PRIM` indicates that the session with IMSNAME1 is the primary session.

  – `PRI` indicates that IMSNAME1 is a primary LU.

- The following is the second IST635I message:

  ```
  IST635I  IMSNAME2  ABKUP-PRI
  ```

  In this message:

- IMSNAME2 indicates that terminal TERM1 is in session with IMSNAME2.
- A indicates that the session is active, that is, not pending.
- BKUP indicates that the session with IMSNAME2 is the backup session.
- PRI indicates that IMSNAME1 is a primary LU.

Displaying the Status of Sessions Connected to Specific Subsystems

The VTAM DISPLAY NET,ID command also allows the network operator to see the status of all sessions on class-1 terminals connected to a specific subsystem in the XRF complex. For example, to see the status of all class-1 terminals that have sessions with the subsystem IMS1, the network operator issues the following command:

```
DISPLAY NET,ID=IMS1,SCOPE=ALL
```

The following message (IST635I) appears for every session that is communicating with IMS1.

```
IST635I   NAME   STATUS   SESSION  ID  SEND  RECV  VRN  TP . . .
IST635I   TERM1   APRIM-SEC
```

In this message:

TERM1 indicates that terminal TERM1 is in session with IMSNAME1.

A indicates that the session is active, that is, not pending.

PRIM indicates that the session is a primary XRF session. All sessions with IMSNAME1 will be primary XRF sessions

SEC indicates that TERM1 is a secondary LU.

Recommendation: If you have many class-1 terminals, do not display all sessions.

Displaying the Number of Primary and Backup Sessions in XRF With USERVAR

The VTAM DISPLAY NET,ID command allows the network operator to see the number of primary and backup sessions on class-1 terminals communicating with XRF subsystems. For example, to display the number of primary sessions in the XRF complex, the network operator issues the following command:

```
DISPLAY NET,ID=IMSNAME1,SCOPE=ALL
```

VTAM responds with this message:

```
IST809I  XRF SESSIONS - PRIMARY = nnnn  BACKUP = 0000.
```

To display the number of backup sessions in the XRF complex, the network operator issues the following command:

```
DISPLAY NET,ID=IMSNAME2,SCOPE=ALL
```

VTAM responds with this message:

```
IST809I  XRF SESSIONS - PRIMARY = 0000  BACKUP = mmmm.
```

Messages IST634I and IST635I follow with information about each session that is communicating with the IMS XRF complex.

An important use of this command is to determine if all the class-1 terminals have backup sessions established (if USERVAR is used). For example, before initiating a planned XRF takeover, an operator should make sure that all the users of class-1 terminals have backup sessions in place. In this case, the operator issues both

DISPLAY commands discussed in this section, compares the number of primary and backup sessions, and does not initiate a takeover unless the two numbers are identical.

Displaying the Current Application Name in XRF With USERVAR

The VTAM DISPLAY NET,USERVAR command allows the operator to see the application name that corresponds to a USERVAR in the USERVAR table. For example, the operator issues the following command:

```
DISPLAY NET,USERVAR,ID=IMS
```

The response to this command is:

```
IST811I USERVAR(S):
IST810I NAME = IMS, VALUE = IMSNAME1
IST8351 END
```

If your installation has defined more than one USERVAR, the network operator can issue the following command to display all the USERVARs and the corresponding application names:

```
DISPLAY NET,USERVAR
```

Maintaining the XRF Complex during Tracking

Maintaining an XRF complex requires that you know how to change surveillance dynamically, how to update system data sets, and how to suspend operations on the active or the alternate subsystem.

Although an end user cannot log on to the alternate subsystem during the tracking phase, the master terminal, secondary master terminal, optional ISC link between two subsystems, and the system console can communicate with the alternate.

You can enter the following commands on the alternate subsystem during tracking:

/DISPLAY

For descriptions of the use of the /DISPLAY command in an XRF complex, see “Displaying Tracking and Other XRF Complex Information” on page 248.

/OPNDST NODE
/CLSDST NODE

Use these commands to open and close VTAM sessions for the master terminal and the ISC link. These commands can reestablish backup sessions for class-1 terminals that have active sessions, if USERVAR is used. Backup sessions are not maintained in XRF systems with MNPS.

If the active IMS has a session that is XRF capable, enter /OPNDST on the alternate subsystem to establish a backup session, if it does not already have one.

In order to use the /OPNDST command successfully for a class-1 ETO terminal, the control blocks must exist and be coupled together, and an active session must exist on the active system. If not, IMS rejects the /OPNDST command, and issues message DFS3862.

/START LINE x PTERM y
/STOP LINE x PTERM y
/RSTART LINE x PTERM y

Use these commands to start, stop, and restart BTAM master and secondary master terminals.

/START NODE x
/STOP NODE x

/RSTART NODE x

Use these commands to start, stop, and restart VTAM master and secondary master terminals.

/START REGION
/STOP REGION

Use these commands to start and stop regions.

/START SURVEILLANCE
/STOP SURVEILLANCE
/CHANGE SURVEILLANCE

For descriptions of the use of these commands in an XRF complex, see “Controlling Surveillance.”

Controlling Surveillance

Use the following command to dynamically stop and start surveillance, and dynamically change surveillance values:

/START SURVEILLANCE LNK | LOG | RDS | ALL
/STOP SURVEILLANCE LNK | LOG | RDS | ALL

Use these commands to start and stop the operation of any (or all) of the three surveillance mechanisms:

- The ISC link
- The system log
- The restart data set

If you operate with all three surveillance mechanisms, IMS can detect a problem as soon as possible and is more likely to initiate a takeover only when it is appropriate. However, you can stop and start surveillance for a brief period to perform a system change, such as to restart a DASD control unit, that might trigger an unwanted takeover.

/CHANGE SURVEILLANCE [LNK | RDS | LOG | ALL] TIMEOUT | INTERVAL

Use this command to change the mechanism, interval value, or timeout value that you established during initialization. If you enter this command on the active subsystem, it affects both the active and the alternate subsystems; if you enter it on the alternate subsystem, it affects only the alternate subsystem.

Recommendation: Enter `/CHANGE SURVEILLANCE` on the active subsystem. However, if you enter it on the alternate subsystem, do not specify an interval or timeout value for the alternate subsystem that is less than the interval or timeout value on the active.

Related Reading: For details about these commands, see *IMS Version 9: Command Reference*.

Updating System Data Sets during Tracking

Because you duplex nonshared system data sets in an XRF complex to eliminate single points of failure, you must coordinate updates to those data sets. Before you make a system change to an XRF-capable IMS subsystem, you must identically update the equivalent data set on the other subsystem in the XRF complex.

Update the following system data sets:

IMS.ACBLIBA	IMS.ACBLIBB
IMS.FORMATA	IMS.FORMATB
IMS.MODBLKA	IMS.MODBLKB
IMS.MATRIXA	IMS.MATRIXB

IMS.TFORMATA	IMS.TFORMATB

You can perform identical updates on both subsystems using an IMS restart or the Online Change Copy utility (DFSUOCU0). Use **one** set of staging libraries for both subsystems; a staging library is not a single point of failure because the online subsystem does not use it. After you have ensured that the inactive data sets on both subsystems are identical in content, use the `/MODIFY PREPARE` and `/MODIFY COMMIT` commands on the active subsystem. After these changes are committed on the active subsystem, the alternate subsystem automatically reprocesses the online change sequence.

Suspending Operations on the Active or Alternate Subsystem

Do not suspend operations on or stop the MVS system on which the active subsystem is running without first disabling surveillance, unless you want to initiate an XRF takeover.

Recommendation: Do not suspend operations on or stop the MVS system on which the alternate subsystem is running without first disabling surveillance and stopping the active subsystem. If you do, the alternate subsystem could interfere with the performance of the active subsystem when you restart the alternate subsystem because it tries to reflect the changes that occurred on the active subsystem in the meantime.

Dumping the IMS Control Region during Tracking

If you specify `QUIESCE=YES` on the MVS `CHANGEDUMP` command, any SVC dump, depending on the timeout values assigned to the `LNK` and `RDS` keywords in DFSHSBxx, might trigger a takeover request. If you specify `QUIESCE=NO`, only a dump of the IMS control region can trigger a takeover. If you are troubled by SVC dumps that cause unwanted takeovers, stop surveillance or specify `QUIESCE=NO`.

Recommendation: To prevent an unwanted takeover while processing an SDUMP:

- Ensure that the XRF takeover interval is sufficient to allow the SDUMP to complete in IMS.
- Ensure that the system data set contains adequate space for capturing information for the dump.

How Commands on the Active Subsystem Affect the Alternate Subsystem

Commands that make resources unavailable to the active subsystem also make these resources unavailable to the alternate subsystem after a takeover. This includes, for example, a `/STOP PROGRAM` command to stop the scheduling of transactions and any command to reassign terminals or stop physical terminals.

Takeover

Most of the processing during takeover takes place in the alternate subsystem. The alternate subsystem detects the normal or abnormal termination of the active subsystem and takes control of its resources and workload.

Planned Takeover

You can apply hardware or software maintenance to either subsystem in an XRF complex, with the following exceptions:

- Do not change the log record format.

- Do not make system definition changes and bring up an alternate subsystem with those changes.
- Do not apply maintenance that would require an IMS cold start in a non-XRF environment.

To apply maintenance to the alternate subsystem: terminate it, leaving the active subsystem without an alternate subsystem for the time that you take to apply the maintenance. Then bring that subsystem back up as the alternate subsystem.

To apply maintenance to the active subsystem: initiate a planned takeover to shift its workload to the alternate subsystem. After you apply maintenance, bring up that subsystem as a new alternate subsystem. Initiate another takeover to return the subsystem to its role as the active subsystem.

Unplanned Takeover

The alternate subsystem can take over the workload of the active subsystem without operator intervention. The following section describes conditions on the active subsystem that can cause a takeover.

Takeover Conditions

In an XRF complex, the alternate subsystem uses log records and surveillance to detect events on the active subsystem that suggest the need for a takeover:

- It reads the log for evidence of an IMS abend.
- It reads the log for evidence of an IRLM failure.
- It detects an absence of signals across the ISC link, absence of new records on the OLDS, or absence of a new timestamp on the RDS.

You can display the surveillance mechanisms and takeover conditions in effect by using the `/DISPLAY HSB` command; see “Displaying Tracking and Other XRF Complex Information” on page 248.

The following conditions can trigger a takeover request:

- A surveillance-detectable failure of the IMS control region
- A total failure of MVS
- A single central processor complex (CPC) failure
- An IRLM failure that causes IMS to call the IMS STATUS exit routine
- A VTAM failure that causes IMS to call the IMS TPEND exit routine

If you set the `SWITCH=` parameter of the DFSHSBxx member in IMS.PROCLIB to `SWITCH=(LNK,LOG,RDS),(IRLM),(TPEND)` (the default), the alternate subsystem requests or initiates a takeover for any one of the following events:

- Both the ISC link and the RDS fail to send signals, and new log records fail to appear after the timeout periods that you specify.
- The IRLM fails and calls the STATUS exit routine.
- VTAM fails and causes IMS to call the TPEND exit routine.

XRF does not take over for the following types of failures:

- A channel or link failure that causes a break in communication between the CPC and the communication controllers
- A failure in the telecommunication network, such as communication controllers, the network control program (NCP), telecommunication lines, or terminals
- An intersystem failure, such as those caused by JES3 or CTC links

- Loss of or damage to an IMS database
- A power failure that affects both CPCs in the XRF complex
- A failure of user catalogs that point to data sets, such as databases

You can establish your own procedures to detect more subtle causes of failures of the active to process transactions. If a failure situation occurs that the alternate subsystem does not detect, you can force a takeover by entering the `/SWITCH SYSTEM` command from either the active subsystem or the alternate subsystem.

The Takeover Flow

Most of the events during a takeover are the same whether the takeover is planned or unplanned, and whether you specify that it is to proceed automatically or require your approval.

If you want a takeover to proceed automatically, specify `AUTO=YES` in the DFSHSBxx member of IMS.PROCLIB. When IMS detects a takeover condition, it does not give you the option of deciding whether a takeover is needed. You receive message `DFS3890I` saying that a takeover was requested, and the alternate subsystem takes over the workload from the active subsystem.

If you want a takeover to proceed only with your approval, specify `AUTO=NO` (the default) in the DFSHSBxx member of IMS.PROCLIB. When IMS detects a takeover condition, you receive message `DFS3869` saying that the active subsystem might have failed. You must decide whether to proceed with a takeover. To initiate a takeover, issue the `/SWITCH SYSTEM ACTIVE [FORCE]` command from the active subsystem if you can. Otherwise, issue the `/SWITCH SYSTEM FORCE` command from the alternate subsystem.

Good communication between operators for the active and alternate subsystems is essential.

Ensuring I/O Prevention

After an XRF takeover, you must ensure that the failing active subsystem cannot update data sets used by the new active subsystem. That is, you must ensure that the new active subsystem is in a state known as *I/O prevention*. If the new active subsystem can write to the databases before the failing active subsystem is prevented from changing this data, data integrity is compromised. I/O prevention can occur in one of the following ways, depending on the cause of the takeover:

- I/O prevention occurs automatically:
  - If IMS fails and MVS continues running
  - If you enter the `/SWITCH SYSTEM` command on the active subsystem and IMS issues abend `U0604`
- If MVS or the CPC fails, ensure I/O prevention by resetting the active CPC.
- For all other causes of takeover, terminate the active IMS subsystem with the MVS `MODIFY` command.

Because there is no communication between the MVS for the active subsystem and the MVS for the alternate subsystem, the operator for the active subsystem must inform the operator for the alternate subsystem when I/O prevention is complete.

You know that I/O prevention is complete in one of the following ways:

- By receiving the following message from the MVS availability manager:

```
AVM006E  TELL OPERATOR AT BACKUP TO REPLY
         "UNLOCK" TO MESSAGE AVM005A.
         I/O PREVENTION IS COMPLETE FOR
         SUBSYSTEM ssid,
         FAILING ACTIVE ELEMENT OF RSE rsename.
```

- By ensuring that the failed IMS control region and DL/I address space have terminated execution
- By ensuring that the CPC has failed
- By ensuring that the CPC has been reset
- By canceling IMS and waiting until all of the address spaces terminate
- By physically switching the DASD that contains the data sets that the active subsystem might access

If the availability manager's address space fails, IMS continues processing normally but takeover processing is degraded. You might choose to continue without the availability manager. You cannot reconnect IMS to the availability manager unless you terminate and restart IMS.

Ending I/O Toleration

Until the failed active subsystem has completed I/O prevention, the new active subsystem is in a state known as *I/O toleration*. You must be sure that I/O prevention is complete before you end I/O toleration. No message appears on the new active subsystem console to tell you that I/O prevention is complete. The only way you can learn that I/O prevention is complete is from the operator of the failed active.

Recommendation: Because the new active subsystem is executing transactions while the failed active subsystem is in I/O prevention, it is better to wait until you are sure that the IMS control region and the DL/I address space on the failed active subsystem are fully terminated than to get out of I/O toleration prematurely.

You can end I/O toleration in one of the following ways:

- By responding `UNLOCK` to the following message:

  ```
  AVM005A  REPLY UNLOCK WHEN I/O PREVENTION
           COMPLETES FOR RSE rsename
  ```
- By entering the `/UNLOCK SYSTEM` command

Manual Control at Takeover

You give an operator control during an XRF takeover for one or more of the following reasons:

- To perform certain tasks
- To make the takeover smoother
- To make sure the takeover is necessary
- To try to correct the problem that caused the takeover request

When IMS issues message `DFS3890I` to initiate the takeover request, the IMS operators must have specific procedures to follow. The IMS operator for the active subsystem first verifies that a takeover is necessary. The message might indicate a condition that does not call for a takeover, such as a link failure or the active subsystem starting a dump. The message appears every 30 seconds as long as surveillance or the system log identifies the error condition to the alternate subsystem. If the active subsystem resumes normal processing, the message no longer appears.

If a takeover is necessary, before allowing the takeover to proceed, an IMS operator might:

- Switch telecommunications lines
- Suspend operations for certain workloads on the alternate subsystem, such as stopping a batch job
- Force TSO terminal users off the alternate subsystem
- Switch a 3274 terminal controller to allow recovery on local terminals
- Switch the databases connected to the active subsystem to the alternate subsystem

To attempt to fix the problem that caused the takeover request, the IMS operator for the failing active subsystem might:

- Restart the CPC
- Terminate the active IMS subsystem
- Respond to a missing interrupt handler (MIH) message on one of the MVS consoles

To proceed with the takeover, the operator for the active subsystem issues the `/SWITCH SYSTEM ACTIVE` command from the IMS console for the active subsystem. This command causes the active subsystem to terminate abnormally with a `U0604` abend.

The operator for the alternate subsystem can request that the takeover proceed by issuing the `/SWITCH SYSTEM FORCE` command The operator for the active subsystem then terminates IMS in the active. When address spaces in the active subsystem are all terminated, the operator for the active subsystem can assume that I/O prevention is complete.

Recommendation: The operator for the alternate subsystem should initiate a takeover only when the active subsystem cannot receive commands.

At this point, procedures for the operators are the same as if the takeover proceeded automatically rather than manually.

Failure during Takeover

If a failure of the alternate subsystem occurs during takeover (regardless of whether it is sharing data), restart (`/ERE`) the subsystem that took the latest checkpoint valid for restart.

Network and End-User Considerations

Because the purpose of an XRF complex is to diminish the disruption from failures and planned changes to end users, you must be aware of how an XRF takeover alters your network and terminal configuration.

XRF Support of Terminals

The support that XRF provides to a particular terminal depends on several factors:

- Whether the terminal uses the SNA protocol
- Whether the terminal uses the logical unit type 6.2 (LU 6.2) protocol
- How you define the terminal on the `UNITYPE` keyword of the TYPE or LINEGRP macros during system definition or ETO logon description
- What priority you establish for the terminal on the `BACKUP=` keyword of a system definition macro or ETO logon descriptor

- Whether the terminal is controlled by a VTAM instance and an NCP that support XRF, if USERVAR is used
- Whether the terminal is connected to a 37x5 Communication Controller
- If the terminal is controlled by BTAM, whether it is non-switched (that is, not dial connected) or on a multi-system line access (MSLA)
- Whether the terminal is channel-attached to the active subsystem or the alternate subsystem
- Whether you specified `BACKUP=(, YES)` for class-1 terminals or `BACKUP=(, NO)` for class-2 terminals
- Whether the VTAM instance that owns the terminal is in the active subsystem's MVS, the alternate's, or in a Communication Management Configuration (CMC)

XRF provides different service after takeover for the three different classes of terminals, depending on their characteristics. For details about the characteristics of terminals in each class, and the extent of XRF support for each class, see *IMS Version 9: Administration Guide: System*.

XRF Support: During takeover, an XRF-capable IMS subsystem manages its three terminal classes in different ways.

Class-1 Terminals: Class-1 terminals are managed differently depending on whether XRF with USERVAR or XRF with MNPS is used.

If XRF with USERVAR is used, IMS asks the Network Control Program (NCP) to switch VTAM sessions for class-1 terminals from their active sessions to their pre-initialized backup sessions and to terminate their active sessions.

If XRF with MNPS is used, VTAM holds the class-1 terminals' sessions open as persistent sessions when XRF takeover begins. The original MNPS ACB closes and the alternate IMS opens its own MNPS ACB. VTAM then reconnects the class-1 terminals' sessions to the new MNPS ACB on the new active IMS.

Class-2 and Class-3 Terminals: If XRF with USERVAR or XRF with MNPS is used, class-2 and class-3 terminals are managed as follows:

- IMS tries to reestablish sessions for class-2 terminals.
- IMS terminates sessions for class-3 terminals, as it would if XRF were not present. Logical unit type 6.2 (LU 6.2) devices are class 3.

Sessions for class-2 terminals terminate at the time of failure, but the alternate subsystem tries to reestablish sessions if:

- Paths between the alternate subsystem and the terminals exist.

  Paths exist for channel-attached terminals connected to other hosts in the network. Paths do not exist for channel-attached terminals connected to the failing active.
- The alternate subsystem can find the host that owns the terminals.

  It does this by specific definition in the Cross-Domain Resource Definitions or by using the VTAM facilities that try to locate the owning system services control program (SSCP) by referring to a default list of SSCPs.
- The owning SSCP is available.

  The alternate subsystem cannot reestablish a session with a class-2 VTAM terminal if the failing active subsystem owns it.

BTAM: During initialization of the new active subsystem, if you have terminals controlled by BTAM, you might receive the following message for each BTAM line:

```
IEC804A  CONTROL UNIT NOT OPERATIONAL.  REPLY: CONT, POST, or DROP
```

To allow IMS to reconnect terminals on these lines, reply `DROP`.

XRF and TSO, Batch, CICS, or Another IMS

During XRF takeover, MVS might swap out swappable work, such as TSO and batch applications, until the new active subsystem is stable. When MVS swaps the work back in, it might run with degraded performance, depending on the demands of the IMS workload. If you want to cancel these jobs on the new active subsystem, wait until takeover is complete, and cancel one job at a time. Processing for the `CANCEL` command takes priority over takeover processing, and could slow takeover.

Nonswappable work during takeover, such as CICS or another IMS, competes with takeover processing, and experiences some degree of degraded performance. MVS tries to give IMS in the new active subsystem priority during takeover, but the nonswappable work might still interfere and prolong the takeover.

Nonswappable work after takeover competes with the IMS interactive workload. Response times for both sets of users might be affected, depending on how heavily system resources are used, and their relative priorities.

Changing the USERVAR

During a takeover, as during initialization, to allow terminals to log on to the currently active subsystem, IMS and network operators must initialize the VTAM USERVAR tables:

- The alternate subsystem automatically issues a `MODIFY USERVAR` command to tell its own VTAM instance to initialize the application name in its USERVAR table.
- If you are using VTAM Release 3.1 or an earlier release, the network operator must enter a `MODIFY` *procname*`,USERVAR` command for any other VTAM instance that owns terminals that log on to the currently active subsystem in the XRF complex.

  This might mean that you must tell operators for VTAM instances that communicate with the XRF complex to enter the `MODIFY` *procname*`,USERVAR` command to change the application name in their USERVAR tables.

  If you use NetView, a command list (CLIST) is available that automatically propagates this change in application name to other VTAM nodes.

Dynamic DL/I Backouts

During takeover, the alternate subsystem in an XRF complex backs out transactions that were in process when takeover was initiated.

For DEDBs on shared DASD, the alternate subsystem performs forward recovery. For DL/I databases on shared DASD, the alternate subsystem reads the database block to which the change log record refers, and then it backs out the change. If I/O toleration has completed, it writes the block back. If I/O toleration is still active, the alternate subsystem creates EEQEs for these blocks and places them in virtual buffers residing above the 16-megabyte line.

Transactions (running on the new active IMS) that access these blocks prior to the end of I/O toleration use these virtual buffers instead of the blocks on DASD. DBRC records the presence of the EEQEs and the IRLM propagates this information to all sharing subsystems. Thus, until I/O toleration ends, these databases can continue to be shared, but the presence of the EEQE prevents a sharing subsystem from accessing these particular blocks.

For DL/I databases on nonshared DASD, the alternate subsystem allows you to defer backout until you switch the DASD and restart the databases. The alternate subsystem schedules the appropriate PSBs for backout. It backs out all databases for each PSB that can access them and stops only the databases that it cannot access.

The alternate subsystem records information necessary for the backout and notifies DBRC. DBRC flags the DBDS record with “Backout required”. You can retry the backout as long as the OLDS containing the pertinent information is available.

Post-Takeover

After you receive message DFS994I (takeover is complete) on the alternate subsystem, you can finish other tasks related to the takeover:

- Easing the workload of the new active subsystem
- Recovering terminal sessions
- Completing dynamic DL/I backouts and starting databases
- Notifying other VTAM instances of the takeover
- Performing diagnostics on the failed subsystem
- Establishing a new alternate subsystem after I/O prevention completes and you take the new active subsystem out of I/O toleration

Easing the Workload after Takeover

You might perform the following tasks to help the new active subsystem adapt to its changed workload:

- Stop batch or TSO work that had been running on the alternate subsystem before the takeover.
- Ease the workload of the new active CPC to improve its performance for online users by stopping certain transactions or applications.
- Inform users of restricted operations (if any).

Recovering Terminal Sessions

If the alternate subsystem fails to reestablish a session with a class-2 terminal, use existing procedures to correct the cause of the failure and reestablish service. Use existing procedures to establish new sessions for class-3 terminals.

If class-1 terminals switch to the alternate subsystem (the new active subsystem), you receive the following messages for them on the master terminal or system console: DFS3859I and DFS3860I saying that all terminal sessions have been switched. If you do not receive these messages, identify the unswitched terminals and issue /CLSDST and /OPNDST commands for them.

If you need to switch many terminals manually, you can use an automated operator program. This program can invoke a set of /DISPLAY NODE, /CLSDST NODE, and /OPNDST NODE commands. See Chapter 13, “Automated Operations,” on page 185 for more information about automated operations.

Enabling BTAM-Controlled Terminals for Automatic Session Reestablishment

If you have terminals controlled by BTAM, either the MVS console for the alternate subsystem or the network console might receive the following message for each BTAM line:

```
IEC804A CONTROL UNIT NOT OPERATIONAL.
        REPLY: CONT, POST, or DROP
```

To allow IMS to reconnect terminals on these lines, reply DROP.

Completing Dynamic Backouts and Starting Databases

Use the /DISPLAY DB BKERR command to determine whether dynamic DL/I backout has failed for any database on nonshared DASD. If it has, switch the DASD to make the database available to the alternate subsystem (using, for example, an IBM 3814 switching unit), and issue a /START DB or UPDATE DB START(ACCESS) command for the affected database to perform the incomplete backout. For shared databases, include the GLOBAL keyword on the /START or route the UPDATE command to all IMSs in the IMSplex.

The /DISPLAY DB BKERR command does not show databases and VSAM KSDSs stopped because of extents by the failing active subsystem in progress during takeover. The following messages inform you of these: DFS2012I, DFS0772I, and DFS0773I. These messages list stopped databases, stopped shared databases, and KSDSs that were not backed out.

Notifying Other VTAMs of the Takeover in XRF with USERVAR

During takeover, if XRF with USERVAR is used, the alternate subsystem issues a MODIFY USERVAR command to change the VTAM application name in its USERVAR table. If you are using VTAM Release 3.1 or an earlier release, network operators in all VTAM domains except the one for the alternate subsystem, including the failed active, must change these entries. The network operators for each of these VTAM instances issue the same command.

When an entry in the USERVAR table changes, VTAM responds with this message:

```
IST813I USERVAR IMS CHANGED FROM name1 TO name2
```

Until the USERVAR is changed for these VTAM nodes, terminals owned by those VTAM instances cannot log on to the new active subsystem.

You might want your operator to delay issuing this command after a takeover until processing on the new active subsystem stabilizes. In fact, if the processing environment for the new active subsystem is inferior to that at the failed active subsystem, you might not want any more terminals to connect to the new active subsystem.

If your installation has installed NCCF, you can use NCCF to communicate the new application name.

Performing Diagnostics after Unplanned Takeovers

If the takeover was unplanned, use existing procedures to determine the cause of the takeover on the failed active subsystem. You might, for example:

- Request service aid information using an MVS MODIFY IMS,DUMP command.
- Perform problem determination.
- Identify the component that caused the subsystem to fail.

- If the subsystem is dumping, let it continue.

Establishing a New Alternate Subsystem

Following a takeover, the new active subsystem runs without an alternate subsystem. Establish a new alternate subsystem by doing one of the following:

- Establish a new alternate subsystem in the CPC that had the failed active subsystem.
- Establish a new alternate subsystem in a different CPC without waiting for problem determination for the failed active subsystem to complete. Given a suitable configuration, this is the faster way to regain XRF capability.

  Restriction: If the XRF complex is sharing databases with another system, do not establish an alternate on a different CPC.

If you want to reestablish your original configuration (active and alternate subsystems in their original CPCs), do so in stages:

1. Bring up the former active subsystem as the new alternate subsystem.
2. Perform a planned takeover.

Termination

The termination phase separates the two subsystems in the XRF complex into independent IMS environments:

- To terminate the alternate subsystem and cancel backup sessions for class-1 terminals, issue the `/STOP BACKUP` command from the alternate subsystem.
- To terminate both the active and the alternate subsystems, issue the enter the `/CHECKPOINT FREEZE` command from the active subsystem. When the active subsystem terminates, it writes a type X'06' record to the IMS log. The alternate subsystem reads this record and terminates.

IMS DBCTL Capabilities

In an IMS DBCTL environment, you can back up your system by:

- Using a second DBCTL subsystem as a standby alternate subsystem
- Using XRF

DBCTL Standby

While one DBCTL subsystem is active, you can start another DBCTL subsystem as a standby alternate. This alternate subsystem does not track the active subsystem (as it would in a true XRF complex), but is a fully initialized IMS DBCTL subsystem that is waiting for a restart command (for example, `/ERE`). This alternate subsystem is a *preinitialized DBCTL environment*.

The preinitialized DBCTL and active subsystem startup procedures must specify the `DBRSE=` keyword with the same string so that after restart the preinitialized DBCTL environment can assume the identity of the failed active. In addition, the active and preinitialized DBCTL subsystems must both specify unique `IMSID=` (or `ALTID=`) and `CRC=` parameters. Otherwise, the second region to start will fail during initialization. Until you issue the restart command for the preinitialized IMS DBCTL subsystem, it cannot process any other commands. After restart is complete, everything is the same as if you had restarted the failed active system. Issuing a restart command for the preinitialized DBCTL subsystem before the active subsystem has shut down causes the restart command to fail and could cause a failure of the preinitialized DBCTL environment.

DBCTL and XRF

An IMS DB/DC XRF subsystem can provide DBCTL service. There are no changes to the XRF functions or to the DBCTL functions in this case.

In order for an XRF takeover to complete in this environment, the CCTL must be able to broadcast a `/SWITCH` command to the XRF active subsystem. The CRC must be unique to each IMS region that specifies it.

Chapter 16. Remote-Site Recovery (RSR)

Generally, remote-site recovery does not change or interfere with normal IMS operations. Most RSR functions and operations are automatic. RSR operational activities at the active site are minimal; operations at the tracking site are quite different from those at the active site.

This chapter describes how to operate RSR-capable IMS subsystems in an RSR complex.

Operating IMS in an RSR Environment

To operate IMS effectively in an RSR complex, you have to understand what tasks to perform during:

- Preparation for tracking
- Tracking
- Remote takeover

You also need to understand:

- Isolated log sender (ILS) processing for gaps in log data
- Catch-up processing
- Online forward recovery
- Restarting new active subsystems

Related Reading: This chapter includes information on the procedures and education that you need to run an RSR complex. For information on these topics from a systems level, see the RSR information in *IMS Version 9: Administration Guide: System*. You should be familiar with the RSR information in *IMS Version 9: Administration Guide: System* before reading this chapter.

Preparing for Tracking

Global service groups (GSGs), service groups (SGs), and registered databases are defined in the RECON data set with DBRC utility commands. You can define multiple GSGs in a single set of RECON data sets (a set of RECON data sets consists of the two active copies and a spare copy).

For a particular GSG, a unique set of RECON data sets exists for each SG. That is, the active and tracking SGs have their own RECON data sets.

The RECON data sets of different SGs need **not** be identical, either physically or logically. The GSG, both of its SGs, and all of the covered databases, must be defined in each set of RECON data sets. In each set of the RECON data sets, one of the SGs is defined as the “local” SG.

The GSG and the SGs are defined using the DBRC `INIT.GSG` and `INIT.SG` commands, respectively. The initial role (active or tracking) of an SG is specified when the SG is defined. IMS subsystems sign on to DBRC specifying the GSG; after a GSG has been defined, the RSR environment can be initialized. IMS subsystems send or receive log data depending on the role of the local SG, active or tracking.

After a GSG has been defined, DL/I databases and DEDB areas for which coverage is desired can be assigned to the GSG. The database or area must be defined in the tracking RECON data set and an initial image copy must be sent to and received at the tracking site.

Recommendation: Each covered (tracked) database and area should be defined in the tracking RECON data set **before** the tracking site receives any log data for these covered databases and areas.

RSR supports the IMS shared-queues environment for active IMS subsystems. A tracking subsystem cannot share IMS message queues with other IMS subsystems, and will abend if started with the shared queue options.

Recommendation: All IMS subsystems participating in sharing IMS message queues should be part of the same service group.

If the active subsystems share the IMS message queues, they must continue to share them after a remote takeover. That is, you cannot change from a shared-queues environment to a nonshared-queues environment during a remote takeover.

Steps Required to Begin Shadowing Data to the Tracking Site

For any given DL/I database or DEDB area, the following steps are required to begin shadowing data to the tracking site:

1. Prepare the tracking site.
   - Define the database or area in the tracking site's RECON data set

     Use the DBRC `INIT.DB` and `INIT.DBDS` commands, specifying the GSG name, to define the database.

     For a DL/I database, all data sets of the database must be defined in the tracking RECON using the same DD names that are used in the active RECON data set. For a DEDB, the areas are assigned to GSGs, rather than the DEDB.

     Only the databases or areas covered by the GSG need to be defined in the tracking RECON data set. Non-covered databases or areas can be defined in the tracking RECON data set, but no log records are sent (if you use the default log filter exit routine, DFSFTFX0) to the tracking site for those databases or areas.
   - Allocate the tracking database or area data sets

     For database-level tracking, create the DBDSs or ADSs. The physical attributes (such as OSAM blocksize or VSAM CI size) of the tracking data sets should be the same as those of the active data sets, although the data set names can be different.
2. Start the active IMS subsystem.

   Be sure to specify the GSG name and `RSR(YES)` in the DFSRSRxx member of IMS.PROCLIB.
3. Initiate coverage at the active site.
   - Assign the database or area to the GSG

     If the database is already defined in the RECON data set, specify the GSG name on the `CHANGE.DB` command for DL/I databases or the `CHANGE.DBDS` command for DEDB areas. The database must not be authorized to any IMS subsystem when you issue the `CHANGE` command.

     If the database is not defined in the RECON data set, use the `INIT.DB` and `INIT.DBDS` commands, and specify the GSG name.

- Create image copy data sets.

  An image copy data set of each DBDS or covered area is required.

  - You can use any of the following: the IMS Database Image Copy utility (without the concurrent option), the Database Image Copy 2 utility (with the exclusive option), or some other method (for example, a pack dump).
  - You cannot use a concurrent image copy, that is, a copy created by the Online Database Image Copy utility (with the concurrent option) or the Database Image Copy 2 utility (with the shared option).
- Send the image copy data sets to the tracking site.
- Begin updating the database at the active site.

  Use the `/START DB`, `UPDATE DB START(ACCESS)`, `/START AREA`, or `UPDATE AREA START(ACCESS)` command to allow the online IMS subsystems to use the database or area.

4. Start the tracking subsystem.
5. Register the image copy data sets at the tracking site.
6. Install the shadow database.
   - If you plan to use database-level tracking, the image copy data set must be applied to any DBTRACK-covered database or area. Use the database recovery utility to apply the image copy data sets to the shadow database data sets. Use the `GENJCL.RECEIVE` command to generate the database recovery utility job steps to apply the image copy data set to the shadow database data set.

     See “Sample Tracking Database Setup” for simple procedures for installing image copies at the tracking site.
   - For non-IMS image copy data sets, you must apply the image copy data set to the shadow database data set using the appropriate method (for example, pack restore). After you apply the shadow database data set, use the `NOTIFY.RECOV` command to inform DBRC that the image copy has been applied to the tracking database data set.
7. Begin shadowing (database-level tracking only).

   The simplest way to begin shadowing databases or areas is to use the `/START` or `UPDATE` command on the tracking subsystem. As part of the start command processing, all database change records created since the image copy was created are applied to the tracking database data sets. This process is called *online forward recovery*.

   Initially, you can specify all shadow databases and areas on one `/START` or `UPDATE` command for the most efficient processing. See Table 66 on page 381 for the appropriate keywords for these commands.

Sample Tracking Database Setup

This section illustrates the procedure for initiating tracking of a DL/I database at the database readiness level (DBTRACK). The DBD name of the sample database is IVPDB2. The database has a single DBDS with DD name DFSIVD2. The remote-site recovery environment has already been established: the GSG is defined, active and tracking subsystems are communicating, and so on.

These are the required steps to initiate tracking of a database (or area):

1. Prepare the tracking site
2. Initiate coverage at the active site
3. Install image copy data sets at the tracking site
4. Begin database shadowing

Prepare the Tracking Site: Figure 22 shows a job stream that defines the database in the tracking RECON and allocates the shadow DBDS.

```
//DEFTDB    JOB   ...
//DBRC      EXEC PGM=DSPURX00,REGION=2048K
//SYSPRINT  DD   SYSPRINT=A
//IMS       DD   DSN=IMSIVP33.DBDLIB,DISP=SHR
//JCLOUT    DD   SYSOUT=(A,INTRDR)
//SYSIN     DD   *
  INIT.DB     DBD(IVPDB2) SHARELVL(3) TYPEIMS GSGNAME(MYGSG)
  INIT.DBDS   DBD(IVPDB2) DDN(DFSIVD2) -
              DSN(IMSIVP33.DFSIVD2) -
              NOREUSE RECOVPD(10) GENMAX(5)
/*
//ALLOC     EXEC PGM=IDCAMS
//SYSPRINT  DD   SYSOUT=A
//SYSIN     DD   *
 DEFINE CLUSTER(                                -
                 NAME(IMSIVP33.DFSIVD2)         -
                 NONINDEXED                     -
                 FREESPACE(10 10)               -
                 RECORDSIZE(2041 2041)          -
                 SHAREOPTIONS(3 3)              -
                 UNIQUE                         -
                 VOLUMES(IMS803)                -
                 CYLINDERS(02)                  -
                 CONTROLINTERVALSIZE(2048)      -
                )                               -
            DATA(                               -
                 NAME(IMSIVP33.DFSIVD2.DATA)    -
                )
 /*
```

Figure 22. Sample JCL for Registering and Allocating Databases at the Tracking Site

In Figure 22:

1. The first step calls the DBRC utility to execute the `INIT.DB` and `INIT.DBDS` commands to register the tracking database and its database data set. When this step completes, the databases are *covered* databases.

   In this example no predefined image copy data sets are used, so no `INIT.IC` commands are needed.
2. The second step calls IDCAMS to allocate the database data set.

Initiate Coverage at the Active Site: Initiate database coverage by:

1. Assigning the database to the GSG
2. Making an image copy of the DBDS
3. Sending the image copy data set to the tracking site

The database must not be authorized to any active or batch subsystems when coverage is initiated. Use the `/DBRECOVERY DB` or `UPDATE DB STOP(ACCESS)` command to unauthorize the database from any online IMS subsystems. You can use the DBRC `LIST.DB` command or the DBRC QUERY API request to determine if any subsystem has authorized the database.

Figure 23 on page 269 shows a job stream that initiates coverage at the active site. This job calls the DBRC utility to assign the database to the GSG (`CHANGE.DB` command) and to generate a second job stream (`GENJCL.IC`) to create both an image copy of the DBDS and the commands required to register the image copy at the tracking site.

```
//COVERDB  JOB  ...
//DBRC01   EXEC PGM=DSPURX00,REGION=2048K
//SYSPRINT  DD  SYSOUT=A
//IMS       DD  DSN=IMSIVP33.DBDLIB,DISP=SHR
//JCLPDS    DD  DSN=IMSIVP33.JCLPDS,DISP=SHR
//JCLOUT    DD  SYSOUT=(A,INTRDR)
//SYSIN     DD  *
  CHANGE.DB DBD(IVPDB2) GSGNAME(MYGSG)
  GENJCL.IC DBD(IVPDB2) MEMBER(TRKSETUP)
/*
```

Figure 23. Sample JCL to Initiate Coverage at the Active Site

The `GENJCL.IC` command uses skeletal JCL member TRKSETUP, shown in Figure 24. The tracking setup member begins with an image copy utility skeletal JCL step. For the IC step, this example uses predefined cataloged image copy data sets. The second step uses the DBRC `GENJCL.USER` command to generate a `NOTIFY.IC` command, which will be used at the tracking site to register the image copy data set.

Recommendation: Use a `GENJCL.RECEIVE` command to receive the image copy data set.

```
//IC%STPNO  EXEC PGM=DFSUDMP0,REGION=800K%PARMCIC
//*
//*                JCL FOR IMAGE COPY.
//*
//SYSPRINT  DD SYSOUT=A
%SELECT  DBDS((%DBNAME,%DBDDN))
%DELETE  (%DBADSAV NE 'AVAIL')
//%DBADDN   DD DSN=%DBDSN,DISP=OLD
%ENDDEL
%DELETE  (%DBADSAV NE '')
//%DBDDN    DD DSN=%DBDSN,DISP=OLD
%ENDDEL
%ENDSEL
//DATAOUT1  DD DSN=%ICDSN1,DISP=OLD
//SYSIN     DD *
%ICSYSIN
/*
//***********  END OF IMAGE COPY SKELETAL JCL MEMBER
//GENIC  EXEC  PGM=DSPURX00,REGION=2048K
//*
//*            JCL FOR GENJCL.USER TO GENERATE A NOTIFY.IC COMMAND
//*
//SYSPRINT  DD  SYSOUT=A
//JCLPDS    DD  DSN=IMSIVP33.JCLPDS,DISP=SHR
//JCLOUT    DD  DSN=TSOID.U.CNTL(REMOTEIC),DISP=OLD
//SYSIN     DD  *
  GENJCL.USER MEMBER(NTFYIC) -
         USERKEYS((%DBD,'%DBNAME'),(%DDN,'%DBDDN'))
/*
```

Figure 24. Sample Image Copy Skeletal JCL Member TRKSETUP

Figure 25 on page 270 shows the skeletal JCL used to build the `NOTIFY.IC` and `GENJCL.RECEIVE` commands.

```
//NTFYIC  EXEC  PGM=DSPURX00,REGION=2048K
//SYSPRINT  DD  SYSOUT=A
//JCLPDS    DD  DSN=IMSIVP33.JCLPDS,DISP=SHR
//JCLOUT    DD  SYSOUT=(A,INTRDR)
//SYSIN  DD  *
%SELECT IC((%DBD,%DDN),LAST)
  NOTIFY.IC DBD(%DBD) DDN(%DDN) -
            RUNTIME(%ICTIME) -
%DELETE (%ICTYPE EQ 'SMSNOCIC' | %ICTYPE EQ 'BATCH')
            STOPTIME(%ICSTOP) -
%ENDDEL
            USID(%ICUSID) -
            ICDSN(%ICDSN) -
            VOLLIST(%ICVOLS) -
            %ICTYPE
  GENJCL.RECEIVE DBD(%DBD) DDN(%DDN)
%ENDSEL
/*
```

Figure 25. Sample NOTIFY.IC Command Skeletal JCL Member REMOTEIC

Finally, you need to transmit the image copy data set to the tracking site using a file transfer program.

Install Image Copy Data Sets at the Tracking Site: When the image copy data set arrives at the tracking site, execute the `NOTIFY.IC` and `GENJCL.RECEIVE` job stream that was created at the active site. The resulting job generated by the `GENJCL.RECEIVE` command installs the image copy data set.

Begin Database Shadowing: Use the `/START DB` or `UPDATE DB START(ACCESS)` command at the active site to start the database and begin normal update activity.

RSR in an IMSplex

You can use RSR in an IMSplex. An RSR tracking system can connect to a Common Service Layer (CSL) to enable automatic RECON loss notification or to receive commands from the Operations Manager (OM). See *IMS Version 9: DBRC Guide and Reference* for more details about automatic RECON loss notification.

If an RSR tracking system does not use the OM or automatic RECON loss notification, it does not require an IMSplex, so it does not require a CSL. In this case, the RSR tracking IMS is not able to receive commands from the OM, but can continue to use existing methods for command entry. In the event of a remote takeover, if the RSR tracking IMS is required to take over the workload of an IMSplex, a CSL must be available at the tracking site.

If an RSR tracking IMS connects with the CSL, it registers for IMS commands with the OM using its IMSID. The IMSplex that the RSR tracking IMS joins must be a different IMSplex from the one at the active site.

You cannot use RSR to recover an entire IMSplex, but you can use RSR to recover an individual IMS in an IMSplex.

Tracking the RSR Complex

After the RSR environment (at both sites) has been initialized, tracking begins. The tracking subsystem uses DBRC to record information about the log data that it receives.

As log records are written by an active subsystem, they are also sent to the tracking subsystem, where they are written to SLDSs. These log data sets (tracking SLDSs) are recorded in the RECON data set as they are created. In general, the tracking subsystem receives multiple log streams from the active site, one stream for each active IMS subsystem. Likewise, the tracking subsystem produces multiple output SLDSs, one set for each log stream received.

Tracking Subsystem Log

In addition to the tracking SLDSs, the tracking subsystem has its own log. Like any online IMS subsystem, the tracking subsystem creates OLDSs and SLDSs. The tracking subsystem does not create its own RLDSs because the tracking subsystem does not update databases on its own, but only shadows those database updates done at the active site. All of these logs are maintained by DBRC.

The tracking subsystem's OLDSs are recorded in PRIOLDS and SECOLDS log records, just as is done for active IMS subsystems. Because there are no RLDSs, there are no PRILOG or SECLOG records for the tracking subsystem.

The SLDSs created when the tracking subsystem's OLDSs are archived are recorded in tracking SLDS records (PRITSLDS and SECTSLDS). With the exception of these additional record types, tracking SLDS records have exactly the same format and content as the SLDS records (PRISLDS and SECSLDS) used at the active site. The PRITSLDS and SECTSLDS record types prevent confusion with the PRISLDS and SECSLDS records created for the active subsystem.

Managing the Databases

When an active subsystem requests update authorization for a database, DBRC determines if the database is to be covered by RSR (tracked). If it is, the requesting subsystem must belong to the GSG to which the database belongs. DBRC rejects the authorization request if a covered database belongs to a different GSG.

A DLT tracking subsystem requires exclusive authorization to the shadow databases. If the database is not currently authorized, DBRC grants authorization to the tracking subsystem in exclusive mode. DBRC rejects the authorization request if the database is currently authorized (for example, to a database utility).

The tracking subsystem receives the database update log records from the active subsystem. For an RLT tracking subsystem, the log records are simply written to the tracking SLDS. For a DLT tracking subsystem, the update records are not only written to the tracking SLDS, but they are also applied to the shadow databases.

For database-level tracking, DBRC knows the current state of the shadow DBDSs, that is, DBRC knows which log records have actually been applied to the DBDSs. For example, if a remote takeover occurs while a tracking DBDS is offline, that DBDS is unusable (marked "recovery needed") until all log records received for that particular DBDS while it was offline are applied.

The Database Update Set

Definition: A *database update set* is the collection of all updates made to the database while the database is continuously authorized to one or more IMS subsystems.

DBRC uses a database update set in order to control the tracking of the databases. A database update set begins with the first update to the database after a period with no update activity for the database. This update can be the first one after starting or restarting the active subsystem or it can be the first update after bringing the database online after it had been offline.

A database update set ends when all IMS subsystems stop updating the database. For example, a /DBRECOVERY DB GLOBAL or UPDATE DB STOP(ACCESS) command (routed to all IMSs in the IMSplex) ends a database update set.

Each database update set is assigned a unique update set identifier (USID). DBRC records the USID whenever a DBDS or area is allocated, and store s the USID in the database record in the RECON data set. The allocation records also contain the current USID.

DBDS and Area Allocation

When the tracking subsystem receives a begin-update or end-update log record for a particular DBDS, the tracking subsystem notifies DBRC. Depending on the type of record, DBRC:

- Writes an allocation record in the RECON data set indicating the start of update activity for the DBDS by the active subsystem that wrote the begin-update record.
- Updates the allocation record to indicate the end of DBDS updates. If the active subsystem later begins updating the DBDS again, DBRC creates a new allocation record.

The existence of an allocation record in the tracking RECON data set indicates that the tracking subsystem has received database change records for a DBDS or area (or that they may be on the way in the case of block-level data sharing). DBRC writes the allocation record to the tracking RECON data set before the tracking subsystem actually applies the updates to the shadow DBDSs.

The tracking subsystem informs DBRC that updates are complete when either of the following occur:

- The tracking subsystem receives an end-update log record
- The active subsystem terminates, and the tracking subsystem receives the end of the log and applies all database change records

If for some reason IMS is unable to apply all of the database change records, the tracking subsystem informs DBRC of the last record applied. DBRC saves this position in the allocation record, along with an indicator that updates are incomplete. Thus, DBRC ensures that the DBDS is not used while updates are pending. The tracking subsystem can query DBRC to determine when to begin applying updates for online forward recovery after database tracking resumes.

When DBRC is notified of an image copy, DBRC deletes extraneous allocation (ALLOC) records and updates active ALLOC records to the time of the first log volume necessary for recovery, based on the oldest image copy for the DBDS or area. It is possible on an RSR tracking subsystem for an ALLOC record to be deleted or updated before DBRC applies the changes, for example, if the database was never started or tracking was stopped (suspended). In this case, you need to recover the DBDS or area before you can restart tracking.

Fast Database Recovery Region

IMS Fast Database Recovery (FDBR) regions are not used at the tracking site. FDBR regions can track and recover databases and areas at active sites, but not at the tracking site.

Remote Takeover

DBRC is involved in remote takeover, both at the original active site and the new active site. From DBRC’s point of view, the main effect of a remote takeover is to reverse the roles of the two service groups (SGs): the original tracking SG becomes the active SG, and the original active SG can later become the new tracking SG (if this is possible and appropriate).

There are two types of remote takeover: planned and unplanned. You initiate a planned remote takeover from the active site; the result of which is in an orderly shutdown of the active subsystems, and a known state of the subsystems at both sites. You initiate an unplanned remote takeover from the tracking site; the result of which is in a known state at the tracking site, but the state of the original active site is unknown: transactions may have been processed at the original active site but the log records containing those transactions may never have reached the tracking site.

The remote takeover process at the new active site is the same regardless of the type of remote takeover. The tracking subsystem completes its work and terminates. After all log data has been written to SLDSs and (for databases tracked at the database readiness level) applied to the databases, the tracking subsystem informs DBRC that tracking is complete. DBRC then updates the RECON data set to make the local SG an active SG so that active IMS subsystems can be started at the new active site to complete the remote takeover.

At the original active site, the remote takeover process depends on the type of remote takeover:

- For an unplanned remote takeover, there is no remote takeover process: the active subsystems effectively just disappear.

  However, if the site is intact, you might want to make the original active site a new tracking site. Unfortunately, because the takeover was unplanned, the databases at the old active site are not in a consistent state with the new active site. In order to make the old active site into a new tracking site, you must transport image copies of all covered databases from the new active site. This process is exactly the same as initializing any tracking site for the first time, with one exception: you must use the `RESET.GSG` command to remove all recovery information for the covered databases of the GSG from the original active RECON data set.
- For a planned remote takeover, the covered databases are in a known state and no action is required at the original active site.

  When the last active subsystem terminates, DBRC updates the GSG record in the RECON data set, making the local SG the tracking SG. After this is done, you can start a tracking subsystem at the old active site.

HALDB Online Reorganization

HALDB Online Reorganization (OLR) cannot be initiated on an RSR tracking system. However, if you run OLR at the active site, the tracking system can track the updates involved in the OLR as it does for other databases.

Related Reading: For more information about HALDB OLR support, see *IMS Version 9: HALDB Online Reorganization Guide and Reference*.

Normal RSR Operations

This section describes normal operation for an RSR environment.

The Active Site

Normal operations for an RSR active site are described in the following sections.

Transport Manager Subsystem

You can use five commands to control the operation of the transport manager subsystem (TMS): DEFINE, DISPLAY, SET, START, and STOP.

Three of these TMS commands, DISPLAY, START, and STOP, are used in normal TMS operations. You issue these commands from a SYSIN data set while starting the component or from an MVS console, using the MVS MODIFY command.

Display Command: The TMS DISPLAY command provides information about the definition and status of transport manager resources, such as the GSG name, SG name, and system name. If you need information about IMS resources at the active or tracking site, use the IMS /DISPLAY or QUERY command.

The transport manager only provides communication services for the IMS active and tracking subsystems, so it does not know of either site's activities.

Start Command: The TMS START command activates the transport manager, isolated log sender (ILS) process, or the system, depending upon which keywords you use with the command:

- START TMS starts the transport manager subsystem.
- START ILS starts an ILS task.
- START SYSTEM initiates conversations with the specified TMS subsystem.

Stop Command: The TMS STOP command stops and deactivates the system, conversation allocation retry, or ILS, depending upon which keywords you use with the command:

- STOP SYSTEM terminates conversations with the specified subsystem.
- STOP RETRY stops automatic retry of conversation allocations for a specified TMS subsystem.
- STOP ILS terminates an ILS process.

Related Reading: The TMS commands are described in detail in *IMS Version 9: Command Reference*.

The IMS Logger

The transport manager receives the contents of OLDS buffers each time I/O for the OLDS starts. The transport manager sends the contents of the OLDS buffer to the log router at the tracking site. If VTAM becomes overloaded (indicated by message DFS4020I), the logger suspends sending of OLDS buffers until the next OLDS switch. If other errors occur while sending records, the logger terminates the conversation.

If there is no current conversation and an OLDS is full, the logger tries to re-establish a conversation with the log router; otherwise, the logger resumes a suspended conversation.

Use the IMS /STOP SERVGRP command if you do not want the logger to automatically attempt to establish a conversation for an OLDS switch. Use the /START SERVGRP command to attempt a conversation connection before the next OLDS switch.

If a connection to the tracking site is lost, DL/I batch at the active site does not try to re-establish the connection. The tracking subsystem detects the missing data and uses ILS to obtain it.

Log Filter Exit Routine, DFSFTFX0

The Log Filter exit routine lets you control the amount of log data sent to the tracking subsystem. IMS supplies a default routine, which eliminates diagnostic data and block-padding data.

Related Reading: See *IMS Version 9: Customization Guide* for more information on the Log Filter exit routine.

High-Speed Sequential Processing (HSSP) Support for RSR

In a non-HSSP environment, IMS logs DEDB updates in the X'5950' log record. In an HSSP environment, IMS handles DEDB updates (not including backward references and sequential dependent updates) differently:

- The updates are stored in an image copy data set.
- IMS writes a X'5947' log record, which contains a bit map corresponding to the updated VSAM control intervals (CIs).

  The actual updated data is in the image copy data set and not on the log. For recovery, IMS uses the image copy data set as a log extension. When the X'5947' log record is processed, IMS uses the bit map to get the corresponding data from the image copy data set.

For HSSP with RSR, IMS logs both the X'5947' and X'5950' log records because the tracking site uses the X'5950' records for recovery. When recovery is required at the active, IMS uses the image copy.

In a non-RSR environment, you can choose whether or not to log both records by using a set options (`SETO`) control statement keyword, `LOGALL`, in the DFSCTL data set. IMS ignores this keyword in an RSR environment.

Online Change

The tracking subsystem does not automatically shadow online changes made to active site subsystems. The tracking subsystem does receive the online change log records (unless you filter them out using the Log Filter exit routine).

Recommendations:

- Before making online changes to an active IMS subsystem, copy the changes to the inactive library at the tracking site.

  This ensures that a copy of the changes is available if a remote takeover occurs after an online change at the active site.
- If you have a lot of online changes to make in a short period of time, be sure the online change log records are received and processed at the tracking site before committing the changes at the active site.

  The tracking system issues a message for each committed online change.
- If an online change affects the physical structure of a database, such as changing the logical record length or block size, you should copy the changes to the tracking-subsystem staging library before making online changes and restarting the database at the active site.

  Afterwards, perform online changes to the tracking subsystem.
- Because the active site can change roles and become a tracking site after a remote takeover, you should maintain, at the active site, tracking libraries that reflect online change activity.

Data Sharing

In a data-sharing environment, IMS maintains database integrity for shared, covered databases at the tracking site. Active subsystems perform the necessary activities for support of tracking databases participating in block-level data sharing.

DL/I Batch

DL/I batch jobs at the active site can update databases that are tracked at the tracking site. All of the updates made by the batch job must be transported to the tracking site to maintain data integrity.

Restriction: In order for RSR to transport DL/I batch log data to the tracking site, the batch job must use both logging and DBRC.

You can use the Log Filter exit routine (DFSFTFX0) to filter batch job logs.

If a connection to the tracking site is lost, DL/I batch at the active site does not try to re-establish the connection. The tracking subsystem detects the missing data and uses ILS to obtain that data.

The Tracking Site

Normal operations for an RSR tracking site are described in the following sections.

Log Router

During normal operations, the log router component of the tracking IMS subsystem does the following:

- Accepts and starts conversations
- Receives and stores active IMS log data
- Supports data sharing
- Sends data to database trackers
- Archives SLDSs
- Initiates and participates in milestone processing
- Truncates logs

Accepting and Starting Conversations: The log router, or an active-site component such as the logger or ILS, initiates conversation allocation. As part of its initialization, each component tries to allocate a conversation with its “peer” (equivalent component performing the same or similar function) at the other site. If the peer is not yet initialized, the allocation fails, and the component waits for its peer to allocate the conversation at the other site or for a `/START SERVGRP` command to initiate the conversation.

Receiving and Storing Active IMS Log Data: The IMS logger uses a transport-manager-subsystem conversation to pass active IMS log data to the log router. During processing of the first buffer of log data, the log router opens a pre-allocated SLDS and registers it with DBRC. Thus, one pre-allocated SLDS always exists for each log stream at the tracking site. The log router writes the buffer’s contents into the SLDS. When the first SLDS is full, the log router closes it and notifies DBRC. The log router then opens a new SLDS, tells DBRC about it, and writes log records into the new SLDS.

At any one time, the output from a DBRC `LIST.RECON` command might show multiple tracking SLDSs with no start and stop times; this is normal for the tracking subsystem. All tracking logs have start and stop times when appropriate. IMS ensures that when a remote takeover occurs, all tracking logs have start and stop times.

If you request automatic archiving by specifying the ARCHIVE keyword in the DFSRSRxx member, the closed SLDSs are automatically archived. See “Archiving SLDSs” for more information on archiving the SLDS.

Some types of logged information cause the log router to send requests to DBRC to ensure that the state of the databases is maintained in the RECON data set. This action occurs whenever the log router detects one of the following events in the log data received from the active subsystem:

- Begin update
- End update
- Batch backout
- Timestamp forward recovery

Compressing PRILOG Records: The tracking subsystem receives and manages logs received from active subsystems. The tracking subsystem compresses the PRILOG records in these logs, just as an active subsystem does. However, this compression is independent of the compression done at active sites: an active subsystem’s PRILOG record could fill up and need compression before the tracking subsystem’s PRILOG record, or the tracking subsystem’s could fill up first.

It is important that the tracking subsystem receive current log data, to ensure that data is not compressed by the active subsystem before the tracking subsystem receives it. You can adjust the log retention period for the active subsystem’s RECON data set to reduce the risk of this happening.

Recommendation: Do not record image copies at the tracking site so that the time of the oldest image copy is later than the current routing time. Use the /DISPLAY TRACKING STATUS command to determine the current routing time (the ROUTED-LOG time).

Related Reading: For more information on compressing the PRILOG records or on changing the log retention period, see *IMS Version 9: DBRC Guide and Reference*.

Database Tracking: For a DLT tracking subsystem, additional processing occurs after the log data is stored at the tracking site. The log router passes database change records to the individual components responsible for database tracking. These individual components apply the changes to those databases registered as DBTRACK. See “Database Tracking” on page 279 for more information on database tracking.

Block-Level Data Sharing: The log router merges database update records in creation-time sequence. If log records are missing (that is, a gap exists) from one or more active subsystems for any shared database, the log router continues presenting records to the database tracking components, up to the beginning of the gap in the log. The log router stops all routing of the log data at the gap point, until the gap is filled. The tracking system continues accepting log data from the active site. After the gap in the log has been filled, routing resumes.

Archiving SLDSs: You can request automatic archiving of the tracking SLDSs by specifying the ARCHIVE keyword, with various parameters, in the DFSRSRxx

IMS.PROCLIB member. If you do not specify ARCHIVE in the DFSRSRxx member, the log router does not perform automatic archiving of the SLDSs and you must use offline utilities to archive these data sets.

You can use the GENJCL.ARCHIVE command to archive tracking SLDSs manually when you do not use automatic archiving.

After a successful archive, IMS deletes the original SLDS. The archived copy is the only one available for restart, online forward recovery, or catch-up processing.

When you specify ARCHIVE, you can optionally request that IMS create RLDSs by also specifying RLDSDEF in the DFSRSRxx IMS.PROCLIB member.

Recommendation: Archive SLDSs to a storage medium that is either slower or less expensive than the medium used for the SLDS itself. If you have space limitations for your non-archived data, do not delay automatic archiving.

Use the following keywords on the ARCHIVE statement in the DFSRSRxx IMS.PROCLIB member to control automatic archiving: MINSLDS, MAXSLDS, and MAXPROCS.

MINSLDS
Specifies the number of SLDSs to be archived during any one automatic archive operation for a particular active subsystem being tracked.

When the number of tracked SLDSs not needed to restart an active subsystem after a remote takeover reaches the MINSLDS value, automatic archiving begins. The MINSLDS value specifies the number of tracking SLDS to be copied into a single archive data set.

MAXSLDS
Specifies how many SLDSs are to be maintained before an automatic archive is scheduled.

When the log router has a number of SLDSs equal to or greater than the MAXSLDS value for a particular active subsystem being tracked, it starts an automatic archive. This archive is started even if the SLDSs contain data that is needed for restart of the new active subsystem after remote takeover.

MAXPROCS
Specifies how many automatic archive operations can run concurrently.

If the number of archive operations exceeds the MAXPROCS value, some of the archive operations are queued.

All batch logs (SLDSs) become eligible for automatic archive as soon as they are closed.

Examples:

1. You have nine SLDSs not needed for restart for a particular active subsystem, and you specify:

   ```
   ARCHIVE(MINSLDS(3) MAXSLDS(8) MAXPROCS(2))
   ```

   Two archive operations are started:

   - Three SLDSs are archived because the MINSLDS value is three.
   - Two jobs are started because the MAXPROCS value is two. The other job is enqueued.

- Automatic archiving begins because you have exceeded the threshold set by the `MAXSLDS` value.

  After the SLDSs are archived, you have six SLDSs.

2. You have 16 SLDSs not needed for restart for a particular active subsystem, and you specify:

   ```
   ARCHIVE(MINSLDS(6) MAXSLDS(15) MAXPROCS(3))
   ```

   Three archive operations are started:

   - Six SLDSs are archived because the `MINSLDS` value is six.
   - Three jobs are started because the `MAXPROCS` value is three. The other three jobs are enqueued.
   - Automatic archiving begins because you have exceeded the threshold set by the `MAXSLDS` value.

   After the SLDSs are archived, you have 10 SLDSs.

Related Reading: See *IMS Version 9: Installation Volume 2: System Definition and Tailoring* for more information on the DFSRSRxx member of IMS.PROCLIB.

Milestone Processing: The tracking subsystem records tracking status information on its own log (OLDS) during operation to provide restart capability. The collecting and writing of this information is called *milestone processing* and is performed at user-specified intervals between checkpoints and during tracking subsystem shutdown. The log records written during a milestone are used during tracking subsystem restart to determine the correct point on each log stream from which to resume tracking.

The database trackers participate in milestone processing. It is during milestone processing that database updates are actually written to the databases. You specify the milestone frequency using the `MILEINTV` keyword in the DFSRSRxx member of IMS.PROCLIB.

When IMS log routing is idle (that is, no new records are being routed) for five consecutive milestones, IMS ignores further milestone requests until routing activity resumes.

Database Tracking

When the database tracker receives the first change record for a shadow database or area registered as DBTRACK, it must acquire exclusive authorization for and open the database or area. After DBRC grants authorization for the database or area and the database or area has been opened, the database tracker applies log records from the active subsystems to the shadow database or area.

After DBRC grants authorization to the database tracker, no other IMS subsystem, including all database utilities, can access the database or area until the tracker relinquishes its authorization. The database or area remains open and authorized until an error occurs or until you issue a `/DBRECOVERY` or `UPDATE` command for the database or area. See Table 66 on page 381 for the appropriate keywords for these commands.

The log router periodically calls the database trackers to perform milestone processing. During milestone processing, buffers that contain database changes are written to the shadow databases or areas. A /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command for a DL/I database does not complete until two milestones have occurred.

Online forward recovery (OFR) allows the database trackers to bring databases, which had been unavailable, back to the current tracking state. You can initiate online forward recovery by issuing one of the following commands:

- /START DB
- /START DATAGRP
- UPDATE DB START(ACCESS)
- UPDATE DATAGRP START(ACCESS)

OFR is also used when the tracking subsystem receives an initial or a more current image copy that needs updates applied to it to bring the database up to date.

Databases and areas must have a STOPPED status before the tracking subsystem accepts a /START or UPDATE command. The /DBRECOVERY or UPDATE command might be required in specific situations to give the database or area the STOPPED status. One such situation is if an RLT tracking subsystem is switched to database-level tracking (DLT) with databases and areas registered as DBTRACK. In this situation, you need to issue a /DBRECOVERY or UPDATE command for the databases and areas before starting them for online forward recovery processing. See Table 66 on page 381 for the appropriate keywords for these commands.

When the tracking subsystem tracks several active subsystems, because updates from one active subsystem can arrive before updates from another, the log router can pass log data to the database trackers in a different order than which it was created. In order to maintain database integrity in a data-sharing environment, log router merges and routes the data in creation-time sequence. However, log router might concurrently route log data created by consecutively-executing active subsystems or batch jobs.

As a result, a database tracker can receive updates from one update set before receiving updates for a previous update set. In this case, the database tracker does not apply the updates for any of the update sets. Instead, the tracking subsystem stops the database or area and issues message DFS4045A (“OFR NEEDED” reason) or DFS2960A (reason code 10). In response, issue the /START DB, UPDATE DB START(ACCESS), /START AREA, or UPDATE AREA START(ACCESS) command for the affected database or area in order to allow online forward recovery to apply these updates.

Database Commands for HALDBs at the Tracking Subsystem

The following IMS commands for a High Availability Large Database (HALDB) at the tracking site have some special considerations that do not apply for a non-HALDB:

- /DBRECOVERY
- /DISPLAY
- /START
- QUERY
- UPDATE

When you issue one of these commands, the database name can be a shadow non-HALDB, a shadow HALDB partition, or a HALDB master.

A command that specifies a HALDB partition operates like a command for a non-HALDB in that the states of the HALDB partition include STOPPED, NOTOPEN, or ALLOCS. When you issue a /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command for a HALDB partition, the partition is affected as follows:

- Tracking of changes in the partition is suspended.
- The partition assumes the stopped state, as when you issue the /DISPLAY DB or QUERY DB command.

For the HALDB partition, the tracking-suspended state and the stopped state occur together, even though the two states are slightly different in concept.

When you issue a /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command for a HALDB master, the master and all of its partitions are affected as follows:

- Tracking is suspended for all of the partitions, but the partitions do not assume the stopped state because of the /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command (though they might be in the stopped state for some other reason).
- The master assumes the stopped state, as when you issue the /DISPLAY DB or QUERY DB command.

To start Online Forward Recovery (OFR) for the HALDB partitions after the master has been stopped, issue a /START DB or UPDATE DB START(ACCESS) command for the master. These commands start OFR for only the partitions that were not stopped individually. Note that a partition could be stopped individually by either a /DBRECOVERY DB or UPDATE DB STOP(ACCESS) command or an error. To start OFR for such a partition, enter a /START DB or UPDATE DB START(ACCESS) command for the partition. The rule is that tracking occurs for a HALDB partition only when neither the partition nor its master is in the stopped state.

And note, finally, that because database data is stored only in the HALDB partitions and not in the master, tracking occurs only for the partitions.

Effects of Issuing /DBR, /STA, or UPD Commands on HALDB masters or partitions: The following tables show the results, based on different initial states, of issuing a /DBRECOVERY, /START, or UPDATE command for a HALDB master or a HALDB partition. In the situations illustrated in these tables, tracking is already suspended for partitions shown as STOPPED or NOTOPEN before the command is issued. When the master's initial state is shown as STOPPED, tracking is already suspended for all of its partitions.

Table 38 shows the conditions before and after issuing the /DBRECOVERY or UPDATE command for a HALDB master.

Table 38. Results of Issuing /DBR or UPD Command for HALDB Master

Conditions Before Command		Conditions After Command	
Master	**Partition**	**Master**	**Partition**
Not STOPPED	ALLOCS	STOPPED	NOTOPEN, but not STOPPED Tracking suspended
Not STOPPED	STOPPED, NOTOPEN	STOPPED	STOPPED, NOTOPEN (no change)
STOPPED	NOTOPEN, but not STOPPED	STOPPED (no change)	NOTOPEN, but not STOPPED (no change)

Table 38. Results of Issuing /DBR or UPD Command for HALDB Master (continued)

Conditions Before Command		Conditions After Command	
STOPPED	STOPPED, NOTOPEN	STOPPED (no change)	STOPPED, NOTOPEN (no change)

Table 39 shows the conditions when issuing the same /DBRECOVERY or UPDATE command for a HALDB partition.

Table 39. Results of Issuing /DBR or UPD Command for HALDB Partition

Conditions Before Command		Conditions After Command	
Master	**Partition**	**Master**	**Partition**
Not STOPPED	ALLOCS	Not STOPPED (no change)	STOPPED, NOTOPEN Tracking suspended
Not STOPPED	STOPPED, NOTOPEN	Not STOPPED (no change)	STOPPED, NOTOPEN (no change)
STOPPED	NOTOPEN, but not STOPPED	STOPPED (no change)	STOPPED, NOTOPEN
STOPPED	STOPPED, NOTOPEN	STOPPED (no change)	STOPPED, NOTOPEN (no change)

Table 40 shows the conditions before and after issuing the /START or UPDATE command for a HALDB master.

Table 40. Results of Issuing /STA or UPD Command for HALDB Master

Conditions Before Command		Conditions After Command	
Master	**Partition**	**Master**	**Partition**
Not STOPPED Command rejected	ALLOCS	Not STOPPED (no change)	ALLOCS (no change)
Not STOPPED Command rejected	STOPPED, NOTOPEN	Not STOPPED (no change)	STOPPED, NOTOPEN (no change)
STOPPED	NOTOPEN, but not STOPPED	Not STOPPED	ALLOCS OFR started.
STOPPED	STOPPED, NOTOPEN	Not STOPPED	STOPPED, NOTOPEN (no change)

In Table 40, the /START or UPDATE command is rejected because these commands are attempting to start a HALDB master that is not in the stopped state. When this occurs, message DFS4108A (reason code 52) is issued. No change is made in the condition of the master or its partitions.

Table 41 on page 283 shows the conditions when issuing the same /START or UPDATE command for a HALDB partition.

Table 41. Results of Issuing /STA or UPD Command for HALDB Partition

Conditions Before Command		Conditions After Command	
Master	**Partition**	**Master**	**Partition**
Not STOPPED	ALLOCS Command rejected.	Not STOPPED (no change)	ALLOCS (no change)
Not STOPPED	STOPPED, NOTOPEN	Not STOPPED (no change)	ALLOCS OFR started
STOPPED	NOTOPEN, but Not STOPPED Command rejected.	STOPPED (no change)	NOTOPEN, but not STOPPED (no change)
STOPPED	STOPPED, NOTOPEN	STOPPED (no change)	NOTOPEN, but not STOPPED OFR not done

In Table 41, the /START or UPDATE command is rejected because these commands are attempting to start a HALDB partition that is not in the stopped state. When this occurs, message DFS4108A (reason code 52) is issued. No change is made in the condition of the partition or its master. Also, if the /START or UPDATE command attempts to start a HALDB partition whose master is in a stopped state after the command is issued, OFR is not done. When this occurs, message DFS4108A (reason code 53) is issued. The partition's stopped state is reset, but OFR is not started because of the master's stopped state.

Online Change

Perform online change to the tracking subsystem using the normal /MODIFY PREPARE and /MODIFY ABORT | COMMIT command sequence. Usually, the tracking subsystem is not concerned with database definition changes that do not require taking the database offline at the active site. This means, for example, that adding segment types does not require a matching change on the tracking subsystem for correct database tracking.

The tracking system issues a message for each committed online change. However, processing for the online change can occur some time after you receive the message. The tracking subsystem ensures that online change information is not acted upon until the tracking system can be sure that the online change log information will not be lost due to log truncation.

Because the tracking-subsystem libraries can be supersets of, or subsets of, the active-subsystem libraries, you might have to copy selected changed blocks from the active- to the tracking-subsystem libraries, and perform an ACBGEN to produce a proper staging library. If you maintain a library at the active site that will become the tracking library when the active site becomes a tracking site, be sure to copy that library to the current tracking site.

See “Online Change” on page 275 for other considerations for online change.

DL/I Batch

The tracking site receives batch log data and writes it to SLDSs. The tracking subsystem informs DBRC of the log data set and the identity of any covered databases for which database-update records appear on the log. For a DLT tracking subsystem, the log records are also passed to the individual database trackers.

IMS handles batch logs and online logs in the same ways. When a remote takeover occurs, you must back out any uncommitted changes using the batch backout utility (DFSBBO00) because remote takeover processing does not dynamically back out uncommitted changes from batch logs.

IRLM

You do not need the IRLM for a tracking subsystem.

Normal RSR Termination

This section describes normal termination in an RSR environment.

The Active Site

Terminate IMS active subsystems in the usual way, using the IMS commands `/CHECKPOINT PURGE`, `/CHE FREEZE`, or `/CHE DUMPQ`.

Terminating the Transport Manager Subsystem

You can terminate the transport manager without terminating IMS or conversations associated with IMS. To terminate the transport manager subsystem, use the MVS `STOP` command. Terminating the transport manager has the following additional effects:

- The DBRC instance in the transport manager address space stops.
- The ILS stops.
- The transport manager informs components identified to it that it is stopping.

When DBRC and ILS stop, IMS deallocates all conversations between transport managers and closes the VTAM ACB. Existing transport manager conversations between active and tracking IMS subsystems are not affected by shutdown of the transport manager address space. IMS cleans up component-related resources as IMS components terminate.

Recommendation: You should consider the transport manager a never-ending job, closely allied to VTAM. If VTAM fails or is shut down, the transport manager shuts itself down. Except during VTAM shutdown, you should not terminate the transport manager while IMS RSR work is active or anticipated.

The Tracking Site

Normal termination at the tracking site is described below.

Log Router

During normal termination, the log router:

- Deallocates all current conversations

  The log router deallocates all conversations between the log router and its partners at the active site (the loggers and ILS in the global service group). The log router does not allocate or accept any new conversations after the start of normal shutdown processing. The log router also disconnects from the transport manager.
- Processes all active IMS log data in the system

  When normal shutdown processing begins, the log router can have log data that has been received but not yet stored and processed. In this case, IMS stores all active IMS log data held by the log router in the appropriate SLDSs, and closes and records these SLDSs in the RECON data set.
- Stops online forward recovery processing

When the log router begins its shutdown processing, it stops online forward recovery processing. When the tracking subsystem restarts, IMS resumes any online forward recovery processing where it left off, unless the databases were recovered offline.

- Stops automatic archive

  IMS stops all automatic archive processes immediately during normal shutdown.
- Deletes unused, preallocated SLDSs

  IMS deletes all unused, preallocated SLDSs during normal shutdown.
- Initiates a termination milestone

  During normal shutdown, IMS records a termination milestone.

DL/I Database Tracking

During normal termination, the DL/I database tracker ensures that all queued work is processed and that all altered buffers are written to the shadow databases.

Fast Path Database Tracking

When the database trackers receive the termination milestone from the log router, IMS begins shutdown for Fast Path database tracking. The termination milestone is the last milestone processed by the database trackers. IMS writes all received and committed updates to the shadow databases before termination. IMS ignores any update log records received but not yet committed. The log router passes these ignored records to the Fast Path database trackers during the next restart of the tracking subsystem. These records could also be processed if a remote takeover occurs.

IMS Restart Without Remote Takeover

This section describes restarting IMS without remote takeover.

The Active Site

Starting the active site after a termination, without remote takeover, is identical to restarting a non-RSR subsystem.

The transport manager does not maintain any information across a component failure. Therefore, it does not have restart or recovery functions that apply to component or CPC failure.

The Tracking Site

The tracking subsystem supports both IMS emergency restart and normal restart for recovery from tracking subsystem disruptions. A normal or emergency restart of a tracking subsystem is **not** a remote takeover. Therefore, restarting a tracking subsystem is much like restarting any other IMS subsystem.

To provide restart capability for the tracking subsystem, IMS performs milestone processing to record additional information on the tracking subsystem’s log during normal operation. These log records contain tracking status information about each active log stream being processed to enable the tracking subsystem to continue tracking where it left off prior to restart. Milestone processing occurs at regular intervals between checkpoints and requires less processing than an IMS checkpoint.

When IMS log routing is idle (that is, no new records are being routed) for five consecutive milestones, IMS ignores further milestone requests until routing activity resumes.

A cold start of the tracker erases the tracking subsystem's tracking status. After a cold start, the tracking subsystem does not resume tracking for any log streams for which tracking had not completed before the restart.

Recommendation: You should only cold start the tracking subsystem after first trying warm or emergency restart and only after taking the correct steps in preparation. In fact, for a cold start other than for the first one, you receive message `DFS2942I`, which gives you the opportunity to change the restart command.

To prepare for a cold start, you must first do the following:

1. Make a backup copy of the RECON data set after all active subsystems' logs have been transmitted, received, archived, and routed, and after all active systems have shutdown normally.

   This backup copy becomes the restored RECON data set if the tracking subsystem must be cold started.
2. Delete all tracking SLDSs created after the RECON data set backup was created.

   When the tracking subsystem is cold started with the restored RECON, IMS spends some time catching up with active logs; reapplying the changes to the databases, though redundant, presents no problem.

   After the tracking subsystem is running, IMS restores normal tracking.

Log Router

After the tracking subsystem terminates without a remote takeover, when you restart it, the log router restarts automatically.

If the tracking site was receiving log data from the active site when the tracking subsystem terminated, after restart processing is complete, the log router repositions itself on the log stream that was being processed prior to termination. The log router obtains the log position of all streams that were active at the time of failure from the tracking subsystem's log records.

Definition: The *log position* is the point in the logs at which the log router needs to start routing log records to the tracking components. IMS deletes tracked logs that are unused at the time the tracking subsystem failed as part of restart processing.

DL/I Database Tracking

IMS restores the status shadow databases had prior to termination. For example, a database that was stopped remains stopped.

After the tracking subsystem is restarted, IMS restarts online forward recovery processes that did not complete prior to termination. See "Online Forward Recovery" on page 289 for more information.

Fast Path Database Tracking

The log router reprocesses log records that had been written to the tracked logs prior to termination but that had not yet been committed to the database.

After the tracking subsystem is restarted, IMS restarts online forward recovery processes that did not complete prior to termination. See "Online Forward Recovery" on page 289 for more information.

Online Change

The ACBLIBx, FORMATx, MATRIXx, and MODBLKSx data sets for the active site and their corresponding data sets at the tracking site should be identical before starting the tracking subsystem for the first time. After the first time, you can use IMS online change commands to maintain these data sets.

IMS Abnormal Termination in an RSR Environment

This section describes abnormal termination in an RSR environment.

The Active Site

RSR does not affect IMS subsystem abnormal termination.

The Transport Manager

If an abnormal termination occurs, the transport manager generates a dump using the `SDUMP` MVS macro and notifies IMS components that the TMS is terminating. IMS components that have existing conversations can continue to use them after the transport manager terminates.

You can use the MVS `CANCEL` or `STOP` commands to terminate the transport manager abnormally.

If ILS abnormally terminates, any activity that was in progress also terminates, but it does not terminate the transport manager subsystem.

The Tracking Site

If DL/I database tracking abnormally terminates, any tracking activity in progress also terminates.

If Fast Path database tracking abnormally terminates, all tracking activity that is in progress terminates immediately.

In the case of pseudo-abends, instead of bringing down the entire tracking subsystem, IMS generates a SNAP DUMP and stops the specific database or area. This stop processing includes tracking termination for the database or area, database or area close, deauthorization, and deallocation. Tracking then continues for other databases and areas not affected by the pseudo-abend.

Processing Gaps in the Logs

This section describes the process by which IMS sends missing log data to the tracking site.

The Active Site

Sometimes, IMS cannot send log data to the tracking site at the same time as it writes the data to the OLDS at the active site. For example, link failures and other disruptions on the active subsystem can cause IMS not to send log data. These gaps in log data transmission must be filled. Sending missing log data is called *isolated log transport processing*.

When the log router at the tracking site detects a gap in the log data, it sends a request to the isolated log sender (ILS) to send the missing log data.

ILS is an RSR component that is required at the active site. ILS runs in the transport manager subsystem address space. It is not required at the tracking site, but you can start it there in anticipation of a remote takeover.

Use the START ILS command to start ILS. You can issue it either from a SYSIN data set as an automated process used when you start the transport manager subsystem, or from an MVS console using the MVS MODIFY command.

When started, ILS identifies to the transport manager subsystem, allocates a conversation with the log router at the tracking site, and waits for work requests from the log router. When it receives work requests from the log router, ILS obtains data set information from DBRC and reads and sends the log data to the tracking site. However, if the missing log data is contained in the OLDS that IMS is currently using, ILS cannot send the data until IMS switches to the next OLDS.

Optionally, you can specify concurrent conversations, using the MAXCONV keyword of the START ILS command. ILS uses multiple conversations if there are multiple log data sets that need to be sent.

If there are multiple transport manager subsystems at the active site, you can start an ILS in each TMS address space. However, the log router establishes conversations with only one ILS at a time.

The Tracking Site

It is the responsibility of the tracking subsystem to detect gaps in the log data and to initiate gap fill processing by sending requests to ILS at the active site. Log data that follows a gap cannot be processed until the gap has been filled. You can determine whether gaps exist and monitor gap status by using the IMS /DISPLAY TRACKING STATUS command.

Catch-Up Processing

This section describes catch-up processing.

The Active Site

The active site does not perform catch-up processing.

The Tracking Site

This section describes situations in which the tracking site might have to perform catch-up processing.

Log Router

Sometimes, log records that are currently being routed to the individual database trackers are not the same log records that are currently being received from the active subsystem. This creates a situation in which the trackers must catch up with the log router.

Catch-up processing is necessary in the following sorts of situations:

- Network disruptions (involving, for example, the transport manager subsystem, VTAM, communications controllers, communication lines)
- Network congestion that temporarily causes the logger to suspend sending log data
- Failure of an active subsystem that is involved in data sharing
- General failure of the tracking subsystem

Catch-up processing executes in parallel with other log router processing in order to minimize the impact of catch-up processing on the rest of the subsystem.

As the data arrives at the tracking site during catch-up processing, it is placed in SLDSs by the log router, and each SLDS is registered with DBRC, just as if it had arrived during normal processing.

Catch-up processing can include the following steps:

- The log router records data from tracked SLDSs before a tracking subsystem failure
- The log router records data from one or more tracked SLDSs during catch-up processing
- The log router processes data from main storage buffers received recently from the active subsystem's logger

Performance of Catch-Up Processing When Using Tape: Because of the number of data sets involved in catch-up processing, and because speed of processing is critical, using tape media for tracked SLDSs for a DLT system can result in poorer performance and some unnecessary activity (such as excessive tape mounts) that delays catch-up processing. If you choose to use tape media for your tracked SLDSs, be sure to minimize any operational delay.

DL/I Database Tracking

Catch-up processing greatly increases the database tracking REDO processing rate. To allow the tracking subsystem to handle this increase, you need to define a large number of PSTs for database tracking.

Recommendation: For the tracking subsystem, define a number of PSTs equal to twice the total number of DL/I PSTs used in all of the active subsystems tracked by the tracking subsystem.

Online Forward Recovery

This section describes online forward recovery (OFR).

The Active Site

The active site does not perform online forward recovery.

The Tracking Site

Online forward recovery is the process by which RSR brings shadow databases and areas that are out-of-date back to the current tracking state. OFR is available only for DLT subsystems and only for databases and areas tracked at the database-readiness level (DBTRACK).

Online forward recovery does not use an image copy to recover, as do offline recovery methods, nor does it use change accumulation data sets. Instead, OFR reads the log data on the tracking SLDSs and applies the updates to the shadow databases. After OFR completes for a given database or area, IMS applies subsequent updates by the normal database-tracking process.

If you have some databases or areas at the tracking site that are defined at the recovery-readiness level (RCVTRACK), you can change them to the database-readiness level (DBTRACK) by doing the following:

1. Ship an image copy of the database from the active to the tracking site.
2. Use the DBRC `CHANGE.DB` or `CHANGE.DBDS` command to change the readiness level to DBTRACK.
3. Install the shadow database using the DBRC `GENJCL.RECEIVE` command.

4. Issue one of the following commands to start the database or data group:
   - `/START DB`
   - `/START DATAGRP`
   - `UPDATE DB START(ACCESS)`
   - `UPDATE DATAGRP START(ACCESS)`

   These commands bring the databases and areas up to current tracking level using online forward recovery.

You can also use online forward recovery when:

- You need to recover several databases concurrently.

  You should always plan to run just **one** OFR process for **all** of your databases that need to be recovered, rather than initiating a separate OFR process for each database. Running one OFR process for all databases greatly reduces the overhead for the recovery (for example, the log is read just once rather than repeatedly for each database).

  Only one OFR can process at a time. Each OFR initiated after the first processes in turn when the previous OFR completes, fails, or is terminated by a `/DBRECOVERY DB` or `UPDATE DB STOP(ACCESS)` command.
- You need to bring the databases up to date prior to performing a planned remote takeover.
- You need to schedule regular off-shift work, including bringing RCVTRACK shadow databases defined up to date.

Depending upon how long the database or area has been offline, or how much log data must be processed for online forward recovery, you might choose to send a new image copy of the database or area from the active site before beginning OFR processing. IMS excludes log data created by the active subsystem SLDSs before the image copy from the log input used for online forward recovery processing.

DBRC maintains the tracking status of each shadow database and area, and it knows when a shadow database or area is not up to date.

Use a display command (such as, `/DISPLAY TRACKING STATUS`, `/DISPLAY DB OFR`, `/DISPLAY AREA OFR`, `QUERY DB STATUS(OFR)`, or `QUERY AREA STATUS(OFR)`) to check the progress of OFR. When OFR has caught up to current database tracking, you see a message (`DFS4056I` for full-function databases, `DFS2989I` for DEDB areas) that says the database is started.

To stop online forward recovery, use the one of the following commands to stop the database or area:

- `/DBRECOVERY DB`
- `/DBRECOVERY AREA`
- `UPDATE DB STOP(ACCESS)`
- `UPDATE AREA STOP(ACCESS)`

You can issue these commands for a single database or area. To issue these comands for a group of databases or areas, use the `DATAGRP` keyword. If part of a group is stopped during OFR, recovery continues for the other databases or areas in the group. If the entire group is stopped, online forward recovery stops for the entire group.

Other commands also stop online forward recovery, such as /CHECKPOINT FREEZE. OFR automatically restarts its processing when the tracking subsystem restarts.

Remote Takeover

A remote takeover allows the tracking site to take over the workload of the active site. After a remote takeover, the former tracking site becomes the new active site; you must define a new tracking site for this new active site in the event that another remote takeover is required. A remote takeover can be either planned or unplanned.

Definitions:

- A *planned remote takeover* allows you to transfer all work from the active site to the tracking site. You might do this, for instance, to minimize system downtime during a hardware upgrade or site maintenance, or if you recognize that a disaster is approaching or imminent.
- An *unplanned remote takeover* allows you to transfer all work from the active site to the tracking site, just as a planned remote takeover does. You initiate an unplanned remote takeover when a sustained outage in system availability occurs at the active site, as in the case of a natural disaster.

Planned Remote Takeover

To initiate a planned remote takeover, issue the /RTAKEOVER command from at least one **active** subsystem. All databases must be in a consistent state for a planned remote takeover to succeed. To ensure their consistency, be sure that all active IMS subsystems have shut down without error.

IMS rejects the /RTAKEOVER command if DBRC has a subsystem record that is marked as abnormally terminated. This record indicates that an online system abended but has not been restarted, or that a batch job has abended and batch backout has not been run.

If IMS rejects the command, you need to restart the failed online subsystem, run batch backout for failed batch jobs, or convert the planned remote takeover to an unplanned one (see “Unplanned Remote Takeover” on page 293).

One consequence of the /RTAKEOVER command is that no IMS subsystem can normally sign on to DBRC. DBRC still permits recovery signon in order to allow abnormally terminated subsystems to be restarted and shutdown.

The tracking site receives notification of the remote takeover through information in the transmitted log data from the active site. You must then stop other active online IMS subsystems in the same global service group (GSG) using either the /CHECKPOINT FREEZE or /RTAKEOVER command. Allow any active batch DL/I jobs to run to completion.

When all subsystem and batch termination completes, DBRC changes the SG’s role at the active site from active to tracking. After the SG’s role is changed to tracking, DBRC does not permit any active subsystems to be started at that site.

If you want to recover the message queues in a non-shared-queues environment at the tracking site after the remote takeover, issue one of the following commands on all active subsystems in the GSG before issuing the /RTAKEOVER command:

- /CHECKPOINT DUMPQ
- /RTAKEOVER DUMPQ

Issuing one of these commands minimizes restart time at the tracking site. If you do not use the DUMPQ keyword on either command, the new active subsystem will use a DUMPQ from a prior checkpoint.

See "Sharing Message Queues after a Planned Remote Takeover" on page 299 for information about recovering the message queues in a shared-queues environment.

Recommendation: Ensure that an ILS is running during planned remote takeover. If ILS is not running and the tracking subsystem did not receive all log data from the active loggers, the remote takeover will not complete.

Related Reading: See *IMS Version 9: Command Reference* for more information on the /RTAKEOVER command.

Performing a Planned Remote Takeover

This section describes the steps you must perform to complete a planned remote takeover. At the beginning of this sample takeover, the active subsystem is at site 1 and the tracking subsystem is at site 2.

1. Issue the /RTAKEOVER command.

   During the remote takeover, you see the following messages, in addition to the usual IMS shutdown messages. The messages shown below are displayed at either the active or tracking sites and are also displayed on either the system console or the IMS MTO. The active subsystem (at site 1) issues the following messages: DFS2939I, DFS058I, DFS4036I, DFS4024I, and DFS4078I. The tracking subsystem (at site 2) issues the following messages: DFS4124I, DFS2932I, DFS2913I, DFS2500I, DFS4125I, and DFS4126I. In the following sample message output, the active subsystem is known as IMSA in service group GLEN and the tracking subsystem is known as IMST in service group (SG) DAVE.

   ```
   DFS2939I REMOTE SITE PLANNED TAKEOVER IN PROGRESS
   DFS058I  hh mm ss RTAKEOVER COMMAND IN PROGRESS

   DFS4124I PLANNED TAKEOVER REQUESTED
   DFS2932I DATABASE UPDATES PRIOR TO SYSTEM  IMSA TAKEOVER HAVE BEEN ROUTED

   DFS4036I CONVERSATION ENDING WITH SERVICE GROUP DAVE
   DFS4036I CONVERSATION ENDED WITH SERVICE GROUP DAVE
   DFS4024I STOP SERVGRP PROCESSING (INTERNAL) COMPLETE

   DFS2913I CONVERSATION WITH IMS  IMSA TERMINATED: ACT SYS SHUTDOWN
   DFS2500I DATASET  IMZnnnnn SUCCESSFULLY CREATED|DELETED|ALLOCATED
   DFS4125I PLANNED TAKEOVER IN PROGRESS

   DFS4078I ILS FOR GSG STL HAS LOST CONTACT WITH SG.SYSTEM: DAVE.TMST

   DFS4126I TAKEOVER COMPLETE
   ```

2. List the RSR records in the RECON data set (optional step).

   Check that DBRC has switched the SG values in the RECON data sets for IMSA at site 1 and IMST at site 2. For example, the RECON data set for IMST at site 2 should show the local SG as active.

   The SSYS record should have been removed in the RECON data set associated with IMSA at site 1 and the RECON data set at site 2 should have a SSYS record for the IMSID of the yet-to-be-activated IMSA at site 2, with ABNORMAL TERMINATION set to OFF and TRACKER TERM set to ON.

3. Change the startup parameters for IMSA at site 2.

The tracking subsystem at site 2 had IMSID=IMST and TRACK=DLT; change these (or use PARM1= on the EXEC statement) to IMSID=IMSA (the IMSID of the active subsystem from site 1) and TRACK=NO.

Switch the APPLIDs for all subsystems. APPLID1, which always specifies the active's APPLID, should be changed to the 'site 2' active APPLID. APPLID3 should be changed to yet to be started site 1 tracking subsystem's APPLID. Also check for the presence of USERVAR in the DFSPBxxx member. If a USERVAR was specified for the active system before the takeover, the same USERVAR should be specified for the IMSA active subsystem at site 2.

4. Start the IMSA active subsystem at site 2 using the /ERESTART command.

   The following messages are displayed on the new active subsystem:

   ```
   DFS2946I  SYSID=IMSA HAS ASSUMED THE ACTIVE ROLE TO SERVICE GROUP DAVE
   DFS994I   RSR TAKEOVER VIA *BUILDQ* EMERGENCY START COMPLETED
   ```

   See “Starting Active Subsystems After Remote Takeover” on page 297 for more information.

5. Change the startup parameters for the new tracking subsystem at site 1.

   The IMSA active subsystem at site 1 had IMSID=IMSA and TRACK=NO. Change the IMSID to IMSx (or use PARM1= on the EXEC statement). The new tracking subsystem IMSID does not have to be the same as the old tracking subsystem IMSID, but in the examples below, IMST is used. Also, change TRACK to RLT or DLT, as required.

   Switch the APPLIDs for all subsystems. APPLID1, which always specifies the active's APPLID, should be changed to the APPLID for IMSA at site 2. APPLID3 should be changed to the APPLID for the new tracking subsystem at site 1.

6. Start the new tracking subsystem by issuing the following command:

   ```
   /NRE CHKPT 0 FMT ALL
   ```

   You see message DFS2923I and many DFS2500I messages, in addition to the regular IMS startup messages.

7. Begin tracking by issuing a /SWITCH OLDS command in IMSA at site 2 or by issuing the /START SERVGRP command on the new tracking subsystem at site 1.

   You see DFS4036I and DFS4024I messages as tracking begins.

8. Perform catch-up, if required.

   If START ILS is included in the TMS startup member, catch-up processing is started automatically. Otherwise, issue F TMSA,START ILS(*ilsname*) on IMSA at site 2.

9. Perform online forward recovery, if required.

   Issue one of the following sets of commands on the new tracking subsystem at site 1 to start the all databases and areas:

   - /START DB ALL and /START AREA ALL
   - UPDATE DB START(ACCESS) and UPDATE AREA START(ACCESS)

Unplanned Remote Takeover

To initiate an unplanned remote takeover, issue the /RTAKEOVER UNPLAN command from the **tracking** subsystem.

Whenever possible, do the following before initiating an unplanned remote takeover:

- Terminate active subsystems.

- For an RLT tracking subsystem, shut it down and restart it as a DLT tracking subsystem, then let online forward recovery recover the databases prior to the unplanned remote takeover.
- Allow ILS processing to complete (to fill gaps in the log data).

  After the /RTAKEOVER command has completed processing, IMS does not accept further ILS data, which can increase the likelihood that you will have truncated logs.

Definition: *Log truncation* is the process by which IMS removes log records from tracking SLDSs so that IMS does not apply invalid data to the databases after the unplanned remote takeover is complete. The following situations can cause log truncation:

- If some log records for an active subsystem are missing at the tracking site, IMS truncates the records beyond the missing data.
- When log streams must be merged in creation-time sequence before routing to the tracking components, routing for all streams stops at the point that a gap is encountered on a log stream or when there is no more log data for a stream.

When an unplanned remote takeover occurs in these situations, IMS truncates the subsystems' logs at the point where log routing has stopped. The log router does not delete the truncated data, however, because you might be able to use it, even though IMS cannot.

After IMS truncates a log, IMS issues a message (DFS2936I) with the truncated SLDS names, including the log record ID of the truncation point. You can then look at the data in the log records offline and take appropriate action.

If you specify the NOREVERSE keyword on the /RTAKEOVER UNPLAN command, you must not reverse the remote takeover. The tracking subsystem saves and processes all data it has received from the active site, regardless of whether that data was committed (written to the OLDS) on the active subsystem. In this case, IMS issues message DFS4121I.

If you do not specify NOREVERSE, the tracking subsystem discards any uncommitted data it has received from the active subsystem.

Performing an Unplanned Remote Takeover

This section describes the steps you must perform to complete an unplanned remote takeover. The output shown assumes the active subsystem is known as IMSA in service group GLEN and the tracking subsystem is known as IMST in service group DAVE. In the following scenario, the current active subsystem (IMSA) is running at site 1 and the tracking subsystem (IMST) is running at site 2.

1. Start the remote takeover.

   Issue the /RTA UNPLAN command on the tracking subsystem. You see the following messages on the tracking subsystem as IMSA shuts down. No messages are shown for the active subsystem because its state is unknown (and unimportant) for an unplanned takeover.

   ```
   DFS058I hh mm ss RTAKEOVER COMMAND IN PROGRESS
   DFS4123I UNPLANNED TAKEOVER IN PROGRESS
   DFS4126I TAKEOVER COMPLETE
   DFS994I *CHKPT yynnn/hhmmss**FREEZE*
   ```
2. Check the RECON data set for the tracking subsystem.

   DBRC switches the SG records, for example, GLEN (the old active SG) is now the tracking SG and DAVE is set to ACTIVE and LOCAL.

3. Restart an active IMS subsystem at site 2.

   To restart an active subsystem at site 2, perform the following steps (same as for a planned takeover):

   - Change the startup parameters for the yet-to-be-started IMSA subsystem at site 2.

     The tracking subsystem (IMST at site 2) had `IMSID=IMST` and `TRACK=DLT`; change these (or use `PARM1=` on the `EXEC` statement) to `IMSID=IMSA` (the IMSID of the active subsystem from site 1) and `TRACK=NO`.

     Switch the APPLIDs for all subsystems. `APPLID1`, which always specifies the active's APPLID, should be changed to the APPLID of IMSA at site 2. `APPLID3` should be changed to the APPLID of the tracking subsystem at site 1. Also check for the presence of `USERVAR` in the DFSPBxxx member. If a USERVAR was specified for the active system before the takeover, the same USERVAR should be specified for the new active system.

   - Start an active subsystem at site 2 using the `/ERESTART` command.

     The following messages are displayed on the new active subsystem:

     ```
     DFS2946I  SYSID=IMSA HAS ASSUMED THE ACTIVE ROLE TO SERVICE GROUP DAVE
     DFS994I   RSR TAKEOVER VIA *BUILDQ* EMERGENCY START COMPLETED
     ```

4. Start a tracking IMS subsystem at site 1.

   To start a tracking subsystem at the new tracking site, perform the following steps:

   - Stop the TMS on the no-longer active subsystem at site 1.

     From the MVS console, issue `P TMSA`.

     The TMS has the RECON data sets allocated and you need to redefine and reinitialize them.

   - Reallocate the RECON data sets and database data sets on the new tracking subsystem at site 1.

     Remember to change the SG parameters (the SG is now local), and check the `RCVTRACK | DBTRACK` parameters in the `INIT.DB`and `INIT.AREA` commands.

   - Ship the image copies and restore them.

     Image copies of the database data sets that are going to be tracked must be made and shipped to the new tracking site. The image copies must be registered with DBRC, and if you are using database-level tracking you must generate jobs to restore them to shadow database data sets.

     | Be sure to issue one of the following sets of commands for all databases and areas on the new active subsystem:

     | – `/DBRECOVERY DB ALL` and `/DBRECOVERY AREA ALL`

     | – `UPDATE DB STOP(ACCESS)` and `UPDATE AREA STOP(ACCESS)`

     | When all the image copies have been received, issue one of the following sets of commands to start all databases and areas on the new active subsystem:

     | – `/START DB ALL` and `/START AREA ALL`

     | – `UPDATE DB START(ACCESS)` and `UPDATE AREA START(ACCESS)`

   - Start a TMS on the new tracking subsystem.

     From the MVS console, issue `S TMSAP,SUB=JES2`.

Aborting a Remote Takeover

After the tracking subsystem has successfully shut down (in preparation for remote takeover processing), you have an opportunity to change your mind about the remote takeover and reverse the process. This reversal can only be performed in the operational window following the /RTAKEOVER command (for either planned or unplanned remote takeover) and before any activity that makes the reversal impossible. Any of the following actions commit you to the remote takeover and make reversal impossible:

- Using the NOREVERSE keyword on the /RTAKEOVER UNPLAN command
- Entering an /ERESTART or /NRESTART command on any IMS subsystem at the new active site
- Issuing a DBRC RESET.GSG command at the old active site
- Submitting a DL/I batch job at the new active site
- Submitting a DL/I Batch Backout utility job at the new active site

To reverse a remote takeover, you must issue the DBRC CHANGE.SG command with the NORTA keyword:

- At both sites for planned remote takeovers
- At the remote site for unplanned remote takeovers

The NORTA keyword indicates that the remote takeover is to be reversed and that the service groups are to continue in their current roles (at their current locations).

Before issuing the CHANGE.SG NORTA command wait for a DFS4123 or DFS4125 message saying that the planned or unplanned takeover is in progress. Using this command before the active or tracking subsystem has acknowledged that the remote takeover is in progress does not guarantee that IMS has updated the RECON data set to reflect the proper state and that the active or tracking subsystem recognize the reversal. If you use the CHANGE.SG NORTA command outside of the allowed remote takeover window, IMS issues message DSP0144I.

If you specify the NOREVERSE keyword on the /RTAKEOVER command, IMS does not prevent you from reversing the takeover. However, if you do reverse the takeover, integrity of the log and databases at the tracking site is unknown. IMS issues message DFS4122A to warn you of potential problems.

Remote Takeover Processing at the Tracking Site

During a remote takeover, planned or unplanned, the tracking subsystem performs some specific processing before shutting down. Primarily, the tracking subsystem completes tracking and then tells DBRC to change the service group to an active role. When this is complete, you can start the IMS subsystems that were running at the old active site at the new active site.

The following sections describe the processing at the tracking site during a remote takeover.

Log Router

During remote takeover, the log router suspends processing and performs the following functions:

- Processes all available log data
- Deallocates all existing conversations with the active sites
- Writes all buffers received from the active sites to SLDSs
- Routes log records for DBTRACK databases to the tracking components
- Halts any online forward recovery processing

After remote takeover, you need to run the appropriate recovery utilities for all databases for which online forward recovery did not complete. If you want to use OFR to recover the databases, delay the remote takeover until OFR completes.

- Truncates the log when necessary

  For an unplanned remote takeover, some of the log data might be truncated so that updates that are out of sequence are not applied to the shadow databases.

DL/I Database Tracking

After a remote takeover is initiated, the DL/I database tracker processes all remaining update records routed from the log router, and applies all relevant database updates to the shadow databases.

IMS terminates any online forward recovery activity still in progress. Databases that were in the process of being recovered with OFR require batch recovery before they can be available for transaction processing. See “Online Forward Recovery” on page 289 for more information on OFR processing.

Fast Path Database Tracking

After a remote takeover is initiated, the Fast Path database tracker stops processing just as if the tracking subsystem had been shut down.

Batch DL/I Handling

After remote takeover completes, any batch DL/I work that was not completed or backed out at the old active site must be backed out at the new active site (the old tracking site) using the Batch Backout utility.

Restriction: Do not begin any Batch Backout jobs until you receive the `DFS4126I` message telling you that the remote takeover is complete. The information in the RECON data set is not complete until the tracking subsystem has successfully shutdown after processing the `/RTAKEOVER` command.

Starting Active Subsystems After Remote Takeover

Before starting your active subsystems at the new active site, you need to ensure that the remote takeover, planned or unplanned, completed successfully (as indicated by message `DFS4126I`). You also need to be sure that you have copies of all of the IMS libraries, including the online change libraries (ACBLIBx, FORMATx, MATRIXx, and MDBLKSx), at the new active site.

For each IMS online subsystem at the old active site, you need to start an equivalent IMS subsystem at the new active site. After a remote takeover, use an appropriate IMS restart command to start the new active subsystems, as shown in Table 42.

Table 42. Starting a New Active Subsystem

Takeover Type	To Save TM Messages	To Discard TM Messages
Planned Remote Takeover	`/NRE BUILDQ`	`/NRE CHECKPOINT 0` or `/ERE COLDCOMM` or `/ERE COLDSYS`
Unplanned Remote Takeover	`/ERE BUILDQ`	`/ERE COLDCOMM` or `/ERE COLDSYS`

Restriction: In a shared-queues environment, you cannot rebuild the shared queues using the IMS restart commands. See “Sharing Message Queues after a Planned Remote Takeover” on page 299 for information about restoring the shared-queues environment.

For unplanned remote takeover, you must ensure that the subsystem records are removed from the RECON data set for all occurrences of active site IMS subsystems. To do this, you must restart (/ERESTART) each one, or use the DBRC DELETE.SUBSYS command (along with any required database recoveries and backouts).

Related Reading: See *IMS Version 9: Administration Guide: Transaction Manager* for information on restarting the IMS terminal network at the new active site.

You must allocate DBDSs and DEDBs for RCVTRACK databases and areas and you must recover them using the database recovery utilities or the change accumulation utilities. In this way, you can make these databases and areas ready for work at the new active site. Because these recoveries are generally done serially, recovery could take a long time.

DBTRACK databases and areas should be ready for work at the new active site with no further processing.

After remote takeover, all shadow database copies become master databases. However, conditions that reflect physical database and DBDS errors are not preserved because these errors did not occur at the old tracking site (the new active site). IMS obtains ADS DD names and data set names from the RECON data set rather than from the log.

The indirect list data sets and primary index data sets for all partitions of partitioned HDAM (PHDAM) and partitioned HIDAM (PHIDAM) databases need to be rebuilt using the Index/ILDS Rebuild utility (DFSPREC0) before the partitions are ready for work at the new active site.

Handling In-flight and In-doubt Data

This section discusses how IMS handles in-flight and in-doubt data on a new active subsystem during its first startup after remote takeover.

Handling In-flight Data after an Unplanned Remote Takeover: During the first startup after a remote takeover, IMS backs out in-flight data for databases that were both authorized to the tracker and registered as covered at the time of the remote takeover.

If in-flight data exists for a non-covered and non-authorized database, IMS does nothing for the database.

If in-flight data exists for a non-covered but authorized database, IMS issues message DFS981I indicating the database was stopped due to a remote takeover.

If in-flight data exists for a covered but non-authorized database, IMS schedules a deferred or restartable backout for the database, and issues message DFS981I indicating that the database is stopped due to remote takeover.

Handling In-doubt Data after an Unplanned Remote Takeover: IMS backs out any in-doubt data that exists after a remote takeover immediately after the first startup completes. Thus, it looks exactly like in-flight data to a new active subsystem, and the actions performed, depending upon authorization and covered status, are identical to those described for in-flight data.

The tracker tracks in-doubt extended error queue elements (EEQEs) to protect the data until the creating-system is started after a remote takeover. The data is thus protected from other subsystems. IMS only registers in-doubt EEQEs at the tracking site, because the in-doubt state is a logical condition that can exist at both sites. Write errors are physical conditions that do not usually exist at both sites. IMS only registers write-error EEQEs if they occur on the tracking subsystem.

Handling In-doubt Data after a Planned Remote Takeover: Only two actions are possible for in-doubt data at remote takeover:

- If the in-doubt data exists for a non-covered database, IMS does nothing for that database.
- If in-doubt data exists for a covered database, IMS schedules a deferred or restartable backout for the database, and issues message DFS981I indicating a backout failure.

Recommendation: Because during a cold start, IMS loses all traces of in-doubt data, do not be attempt a cold start if there is the possibility that in-doubt data exists at the time of the remote takeover.

Because IMS maintains in-doubt data across warm and emergency restarts, IMS resolves in-doubt data by backing it out for covered databases only.

Sharing Message Queues after a Planned Remote Takeover

To rebuild shared queues at the new active site (after a planned remote takeover), issue the /CQSET SHUTDOWN SHAREDQ ON command before issuing the /RTAKEOVER command on the active IMS. The /CQSET command tells the IMS Common Queue Server (CQS) to take a structure checkpoint during CQS shutdown. CQS writes the structure checkpoint to the CQS structure recovery data set (SRDS), which can be used to rebuild the structures on another coupling facility (at the remote site, for example). You could also use the remote dual copy feature of 3990-6 DASD to make these data sets available at the remote site.

After the remote takeover, start active IMSs at the remote site.

Restrictions:

- You cannot use the /ERE BUILDQ command to rebuild the shared message queues. In a shared-queues environment, IMS ignores the BUILDQ keyword.
- You cannot rebuild shared queues after an **unplanned** remote takeover.

Starting a New Tracking Subsystem

Sometime after a remote takeover, you might want to start a tracking subsystem at the old active site or at another site. If you want to use the old active site, the first thing to do after a remote takeover, especially an unplanned one, is to return the operating environment of the old active site to a stable environment, if possible.

After a planned remote takeover, you can start a tracking subsystem at the old active site (or other site) on any CPC that has access to the RECON data sets used by the old active sites. Although not required, you might have the tracker use the same SSID as any one of the old active subsystems. No extra steps are necessary to begin tracking the new active subsystems.

To start a new tracking subsystem at the old active site after an unplanned remote takeover, do the following:

1. Issue the DBRC RESET.GSG command, or rebuild the RECON data set as if you were doing an IMS initial installation. If you use the RESET.GSG command, you also need to use the CHANGE.SG command to change the local role to tracking.

2. Obtain image copies of the databases to be tracked from the new active site, and install them as you did for the old tracking site.
3. Cold start a tracking subsystem, using the `FORMAT ALL` keywords on the `/NRESTART` command.

If you rebuild the RECON data set after an unplanned remote takeover, the new tracking subsystem asks for **all** log data that was created for the GSG. This includes log data tracked before the remote takeover. If you issue the DBRC `RESET.GSG` command, IMS only requests log data after the remote takeover. In either case, if the new active site has been running for a long time before you are able to establish a new tracking site, the tracking subsystem could ask for weeks or months of data.

Recommendation: To prevent the tracking subsystem from requesting lots of data, do the following before reinstalling the tracking subsystem (see “Steps Required to Begin Shadowing Data to the Tracking Site” on page 266):

1. Shut down all active IMS subsystems.
2. Issue the DBRC `DELETE.GSG` command.
3. Issue the DBRC `INIT.GSG` and `INIT.SG` commands to redefine the GSG.
4. Issue the DBRC `CHANGE.DB | DBDS` command to register the databases and areas as covered.
5. Restart the active subsystems.

By following the previous five steps, the IMS tracking subsystem does not request data created before the `DELETE.GSG` was issued.

When the new tracking subsystem has caught up and is tracking normally, you can initiate a planned or unplanned remote takeover, in the same way as described in “Remote Takeover” on page 272.

Discontinuing Use of RSR

If it is necessary to discontinue testing of RSR, yet continue testing the active IMS subsystem, you can do either of the following (assuming the same IMS subsystem is to be used):

- Continue without the tracking subsystem, with no change of parameters.

  You can continue without the Transport Manager Subsystems as well.
- Restart the active subsystem with a new RSR member, specifying `RSR(NO)`.

  In this case, there will be authorization failures against the RSR-tracked databases when they are accessed. To avoid these failures, issue the `DELETE.GSG` command.

Coordinating Recovery with DB2 in a Data Sharing Environment

In data sharing environments where XRC transmits DB2 data to a remote site, RSR provides support for coordinating disaster recovery for IMS and DB2. This support allows IMS and DB2 logs to be synchronized and ensures that IMS and DB2 databases are consistent if a disaster occurs.

XRC is a separate hardware function that tracks DB2 logs. Likewise, RSR tracks IMS logs. RSR coordinates with XRC so that both IMS and DB2 logs are tracked and can be synchronized. To request XRC tracking, in the DFSRSRxx

IMS.PROCLIB member, you specify the session ID of the XRC session that is shadowing the DB2 volumes. As a result, the RSR tracking subsystem processes and synchronizes with XRC processing.

XRC marks all updates with a timestamp on the volumes in the XRC session and uses the timestamps to control updates to XRC-managed secondary volumes. A timestamp called *XRC consistency time* is used to indicate what updates have been applied to the secondary volumes.

When engaged in XRC tracking, the IMS routing point is kept behind the XRC consistency time. The log router maintains the last known XRC time. When the log router encounters a log record that has a timestamp greater than the XRC time, the log router gets a new time from XRC. If the XRC time has not changed, routing will be suspended until a new XRC time has been obtained. Repeated attempts to obtain a new time will occur.

Enabling XRC Tracking

To enable XRC tracking, specify the XRC session ID as a startup parameter for the IMS RSR tracking subsystem. This XRC session must shadow the volumes containing the active logs and BSDSs for the DB2 systems that need to be synchronized with the IMS systems. You specify the XRC session ID in the DFSRSRxx IMS.PROCLIB member used by the RSR tracking subsystem.

Related Reading: For the details of the DFSRSRxx IMS.PROCLIB member, see the *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

Stopping XRC Tracking

The tracking system does not know when to stop XRC tracking. For example, it cannot determine if the DB2s have shut down. It has no knowledge of what data is on the XRC volumes. Its job is to keep the IMS routing point behind the last known XRC consistency time until it is explicitly told to stop XRC tracking.

You may encounter situations when it is desirable to keep IMS RSR tracking running even though the IMS tracking logs will become out of sync with the DB2 tracking logs. For example, if there is a problem with the XRC session, you might want to continue IMS tracking so that, when the XRC session is available, only the XRC data needs to be re-synchronized. To stop XRC tracking, issue the `/STOP XRCTRACK` command.

Resuming XRC Tracking

After XRC is stopped, it needs to be resumed. To resume XRC tracking, issue the `/START XRCTRACK` command. Restarting XRC tracking causes the re-synchronization of IMS tracking with DB2 tracking.

Note: `/START XRCTRACK` **cannot** be used to enable XRC tracking. If XRC parameters were not specified in the DFSRSRxx member when the tracking subsystem was started, then XRC tracking is not available and a `/START XRCTRACK` command will be rejected (message `DFS4110I` will indicate that the command is not valid).

Routing Delays Caused by XRC Tracking

Message `DFS4035A` is issued to alert you that routing has been unable to proceed for a short period of time (approximately 2 minutes) because the XRC consistency time has not changed. When approximately 10 minutes have passed and routing is still suspended, the message is issued again and is repeated at regular 10 minute intervals until the consistency time has advanced.

Routing is also suspended when the tracking subsystem's request to XRC to get the consistency time fails with an error return code. Check the status of the XRC session and correct the problem. If the applications that were updating the XRC volumes have terminated, issue the `/STOP XRCTRACK` command to allow routing to proceed without considering the XRC consistency time.

Gap Processing

When XRC tracking is enabled on the tracking subsystem, log routing stops when a missing log gap is detected or when there is no conversation between the log router on the tracking subsystem and an isolated log send (ILS) component at the active site.

- If there is a missing log gap, the log router acts on the presumption that the missing IMS log needs to be synchronized with the XRC volumes (and with other IMS logs that are being received), so routing is suspended.
- If the log router is not in communication with ILS, routing is suspended because there could be missing log gaps that the log router is not able to detect.

The tracking system should always have a conversation with ILS so that gaps are filled quickly and the tracking system stays current with the active systems. If there is more than one transport manager subsystem (TMS) at the active site, ILS should be started in each so that the log router can quickly establish a conversation with another ILS in the event that the ILS to which it is connected fails. Use the `/DISPLAY TRACKING STATUS` command on the tracking subsystem to determine if there is a conversation with ILS.

Shutdown Processing

When the tracking subsystem processes a shutdown request and XRC tracking is not enabled, routing of the received log data is not immediately terminated. Rather, the log router routes the log streams as far as it can, until it reaches the end of the received log data or reaches a gap in the received log data.

If the tracking subsystem processes a shutdown request and XRC tracking is enabled, routing might not be able to proceed as far as it might have compared to a tracking subsystem that is not performing XRC tracking. After the log router has received the terminate request, routing will terminate as soon as an XRC query does **not** return a timestamp that allows routing to proceed. Otherwise, the tracking subsystem could be delayed indefinitely. For example, if activity on the XRC volumes is low or intermittent while RSR is producing log data at a steady rate, the XRC consistency time could remain constant for long periods of time and log routing would not be able to route all of the received log data.

When the log data is not completely routed, the remote site is not as current as it could have been. When the tracking subsystem is restarted, log routing will resume where it left off, provided that the XRC consistency time allows routing to proceed.

Restart Considerations

When the tracking subsystem is restarted or is cold started with the routing position restored from the position data set, it will note the current and previous status of XRC tracking and take action as appropriate:

- If the previous tracking subsystem was performing XRC tracking but XRC parameters are no longer present in the DFSRSRxx member, message `DFS2919A` will be issued to alert the user that IMS tracking is no longer being synchronized with XRC shadowing. XRC tracking will not be done.

- If XRC tracking was stopped by a /STOP command on the previous tracking subsystem and was not resumed, message DFS2920I will be issued and XRC tracking remains stopped. A /START XRCTRACK command must be entered to cause XRC tracking to be resumed.

Unplanned Remote Takeover

During an unplanned remote takeover, any logs that did not route to completion (because of log gaps or issues related to the XRC consistency time) are truncated at their current routing point. If this truncation occurs, the message DFS2933I is written to the system console (and also written to the MTO for a DB/TM or DCCTL tracking subsystem). This message includes a timestamp that represents the truncation point. You should capture this timestamp and specify it in the conditional restart records for the DB2 systems.

Unplanned Takeover Procedure

Follow these procedures after a decision has been made to perform an unplanned remote takeover.

1. XEND or XSUSPEND

   If the XRC session is still active when you wish to initiate an unplanned takeover, end the session by issuing the XRC TSO command, XEND, or suspend it by issuing the XRC command, XSUSPEND.

2. XRECOVER

   Issue the XRECOVER command on the remote system to complete the updating of the secondary XRC volumes. XRECOVER also labels the secondary volumes with the volume serial numbers of the primary volumes.

3. /RTAKEOVER UNPLANNED

   Enter an /RTAKEOVER UNPLANNED command on the tracking subsystem. To allow the tracking subsystem to route the receive log data as far as possible, the /RTA UNPLANNED command must be issued after recovery (XRECOVER) of the XRC volumes has completed.

If you enter the /RTAKEOVER command before recovery of the XRC session has occurred, the tracking subsystem, shutting down for unplanned takeover, may terminate routing prematurely.

Planned Remote Takeover

The processing of a planned remote takeover for the tracking subsystem is the same whether or not XRC tracking is enabled. However, if routing encounters delays while waiting for a more current XRC consistency time, shutdown of the tracking subsystem will be delayed. In a normal scenario where the XRC session is active and the consistency time is advancing frequently and more or less steadily, there should be little or no delay in the shutdown.

Part 3. Developing Operating Procedures

Chapter 17. Developing Operating Procedures

After you have made basic design decisions about operations and recovery, you are ready to implement these decisions. Primarily, implementation takes the form of developing procedures for operating IMS.

The importance of developing good procedures cannot be over-emphasized. The installations with the most success are those with well-written, well-maintained, well-tested, and well-understood procedures.

Establishing procedures for operating IMS is not a simple task. Because each installation has unique requirements and resources, no single “cook-book” approach to operations is possible. This part gives you some basic guidelines for establishing procedures.

Specifically, this chapter looks at:
- The groups of people involved in operating IMS
- Designing record-keeping procedures
- Designing operating procedures
- Establishing maintenance procedures
- Setting up standard JCL
- Testing procedures

Operations Personnel

Many people are involved, directly or indirectly, in the operation of a successful IMS system. To help you understand who needs operating procedures and what it is they are responsible for operating, this section looks briefly at the various functions performed by these people. It also gives you an understanding of the context in which MTOs operate the system, and end users or remote terminal operators use it. This is important because this chapter and the next describe the procedures you develop for the MTO and end user.

An IMS DBCTL environment includes no IMS terminals, so the DBCTL operator must use an MVS console. Throughout this section, this DBCTL operator is called the IMS MTO.

The following groups of people represent functions often defined at an installation:
- **OS/390, MVS, JES system operators**. This group is responsible for operating OS/390, MVS, and JES.

  If the IMS MTO does not have access to an MVS console, you need to establish procedures so the MTO can communicate with the MVS operator in case of a failure. This is particularly important when the MTO is physically distant from the MVS console.
- **Network support group**. This group is responsible for operation of the network, including: VTAM, physical telecommunications lines, and modems.

  You must establish procedures for communication between the MTO and network support personnel for initializing the online system and isolating and correcting errors related to the IMS network.
- **DASD and tape pool operators**. This group is responsible for managing tape and disk media.

The MTO must communicate with DASD and tape operators when mounting archive tapes, and when creating image copies, change accumulations, and so forth. Communication is also required when the MTO is analyzing and correcting I/O errors.

- **User liaison group**. This group is the primary interface for the end user.

  The user liaison group needs procedures for answering user questions relating to both IMS and individual application programs. The user liaison group also needs procedures for problem determination and correction, and for communicating with the MTO when unable to solve a problem. The user liaison group might also need to communicate with application support groups when a user problem relates to a specific application program rather than to the IMS subsystem.
- **Application support groups**. This group develops and maintains IMS application programs.

  You need to identify representatives for the support of each application running on the IMS subsystem. You should give the user liaison group and the IMS MTO procedures for contacting this group in case of application problems.
- **IMS system administrator, system programmer, or recovery specialist**. This person (or group) is responsible for installing IMS and designing the system configuration.

  There are certain error conditions the MTO cannot correct. For such errors, the MTO needs instructions on how to contact the IMS system administrator, system programmer, or recovery specialist. This person (or group) will help resolve the more complex recovery problems.

Figure 26 on page 309 shows the possible interfaces between these groups. Depending on the size of your installation, some of these functions might be performed by the same person or group. In large installations, the same function might be performed by many different people. The IMS MTO, for example, might have access to an MVS system console. This access would allow the MTO to enter MVS START commands.

It is important that you identify the operational requirements for IMS, and identify the people responsible for performing the various operations tasks. After assigning responsibilities, you are ready to set up the procedures necessary to coordinate the operations task.

In this book, operations procedures are classified into two types:

- Those for operating IMS, which are generally performed primarily by the MTO and secondarily by the system administrator or recovery specialist
- Those for use by an end user

This chapter describes the former. Developing procedures for an end user is described in Chapter 18, “Developing End-User Procedures,” on page 339. Developing procedures for an end user is not applicable in an IMS DBCTL environment, which has no DBCTL terminal end users. References to end users for a DBCTL environment actually refer to the CCTL end users.

Figure 26 on page 309 provides an overview of the relationships between the functional groups involved in the operation of an IMS system.

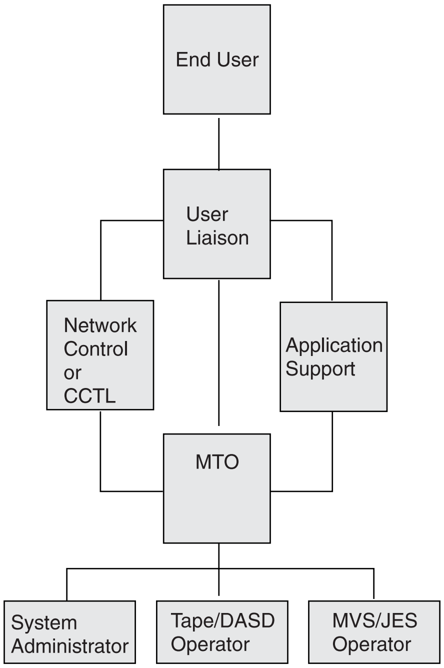


Figure 26. Relationships among Operations Groups

Establishing Operating Procedure Documents

This section addresses the overall strategy for developing procedural control of an online IMS system. The topics are:

- Establishing operations interactions
- Planning for procedure requirements
- Planning the content of procedures
- Setting the level of operator control

Establishing Operations Interactions

The online IMS environment should be transparent to general business procedures. It is probably only part of the data processing services. Your procedures, to some degree, reflect your installation's organization. This section indentifies some of the groups of personnel referred to in later sections.

Definitions of Support Groups

An *IMS operations group* must implement procedures and use tools, often specified by system programmers, that originate from requirements for database and data communication administration. The requirements for scheduling jobs, handling tapes and direct access devices, and producing printed output are developed in conjunction with general system operations procedures.

The *system operations group* provides OS/390 and MVS support, and also often supports the hardware. The *software support group* provides support for the operating system and the language compilers. The *applications support group* provides support for the application programs, and is represented by user liaison and development maintenance groups.

The *user liaison group* is of special importance. Most of the problems that end users experience are application-program dependent. The user liaison group, acting as first point of contact, can answer such questions, but operations cannot. The user liaison group can also be the channel for problem resolution. They can initiate problem tracking, and their interpretation of error events can help all end users. The user liaison group analyzes problems to determine if they are application-program related, procedural, hardware, or system software. They can contact the master terminal control group regarding network or operations problems, or interact with application-program development support for programming errors. Very often, they are key personnel in any decisions affecting recovery, because they must interpret the actions taken on behalf of end users.

Defining Operations Contacts and Responsibilities

Some suggestions for assigning control responsibilities are:

- Have a user liaison group answer day-to-day application-program questions, track problems, and monitor service.

  Make the user liaison group the primary contact for the end user and for IMS operations. Encourage the user liaison group to be part of recovery decisions.
- Have a database-administration contact for recovery, design change evaluation, and database monitoring and analysis.
- Shield the MTO from most end-user inquiries so that the MTO can concentrate on availability, system status, and production statistics. Use other supervisory terminals to distribute the workload. Take advantage of automated operator programs to initiate MTO commands and reduce operator tedium.
- Establish system operations contacts for VTAM operation and hardware problems.
- Have a security contact for access problems. Other duties could be password and security maintenance, as well as violation monitoring.
- Have a monitoring and performance analysis coordinator who draws upon specialists for detailed studies.

These suggested contacts might only be suitable for prime shift or peak production hours. You should plan a contact list for other times, such as production cycle overruns and off-shift periods.

Planning for Procedure Requirements

The established operating procedures reflect many decisions and responsibilities. Development of the procedures takes place during the majority of the application implementation stages. The content of the procedures draws upon several other administration tasks:

- The online IMS design as reflected in system definition
- The variety of application operation requirements
- The database administration procedures

Table 43 on page 311 summarizes the tasks involved in developing a procedural document, relates them to subtasks, and briefly indicates the other groups of personnel involved. The tasks toward the bottom of the figure (starting with "Testing") are concerned with refinement of the procedures.

Table 43. Elements for Establishing Operation Procedures

Development Task	Operations Procedure Subtasks	Related Support Group
Initial structure	Plan terminal operator information Verify IMS message use Plan for automated operator Plan organization and contacts	User liaison (System support) Development
Application requirements	Plan scheduling Plan database availability Assess operator intervention Plan output handling	User liaison and DBA DBA User liaison and DBA Operations
Network control	Plan startup terminals Coordinate operator responsibilities Plan phased connections Plan alternate connections	User liaison and DBA Operations Operations Operations
System definition	Assess restrictions for MTO	Operations
Security	Design procedures Assess operator requirements	User liaison Security control
Administration	Establish system log control Plan error recording Plan network control desk	Operations Operations Operations
Problem procedures	Plan resource recovery	DBA and Development
Monitoring procedures	Plan IMS Monitor use Plan statistics gathering	Performance support User liaison
Recovery procedures	Decide on operator control Decide on problem analysis	Development Support groups and DBA
Testing	Respond to feedback	Development
Master terminal operator training	Respond to feedback	User education
Production control	Audit activities Respond to feedback	Operations User liaison
Modifications to applications	Respond to application-program changes Make tuning changes	Development Performance support

Planning the Content of Procedures

The operating procedure includes four types of instructions:

- The sequence of events for normal operation
- The information to be recorded
- The responsibilities and degree of autonomous action for operators
- The steps to be taken for error incidents or recovery action

Establish the pattern for normal processing by accumulating and combining the different requirements of both end users and databases. This topic is explained in greater detail in “Planning Availability of IMS Service” on page 318.

The information the operators should record primarily concerns recovery actions; however, they should also record events that occur during the production cycle (in a form that can be audited). Many installations use a run book or system operation log for recording events during normal operations.

Whom you choose to have operational control strongly influences your procedure documentation. The instructions can vary a great deal, depending on their knowledge of IMS operation and the application-program logic. Based on your needs, you can train your MTO in one or more of the following areas:

- General operations
- Control of an application on a dedicated system
- Service for multiple applications

The arrangements you make to handle error or recovery events are also important in your control of the online IMS system. A knowledgeable operator can restart IMS and fall back to a valid checkpoint. If the operator suspects database damage, he or she can involve personnel from database administration, or he or she could schedule the required backout or change accumulation jobs.

A suitable document to guide the actions of MTOs should have several sections:

- An introduction

  This section gives an overview of IMS operation (both generally and specifically for your installation), how the network is controlled, and how the master terminal is used. The intended audience is new or substitute operators.
- Operating procedures

  This section shows the normal progression of events, but also provides supplementary instructions for error conditions. You can organize this section as a series of flowcharts.
- Commands reference section

  This section shows syntax and recommendations for using IMS commands and other commands as appropriate. Include only those command keywords and parameters that you consider necessary for situations the MTO should handle without assistance from other groups. You might also want to include customized explanations of selected IMS messages to guide the MTO's actions.
- Configuration tables

  This section contains descriptions of the IMS and system configuration. These descriptions help the operator to visualize the resources under his or her control. For example, you can include a chart of terminal connections related to application-program use, or representations of the databases that are online (together with the logical dependencies). You could also display larger copies in the operations area.
- Supplementary procedures

  This section contains procedures not covered in earlier sections. With any large system, you can make alternative configurations available. This section can also contain recovery procedures indexed by symptom.
- A chronological record of events

  This section lists the various events an MTO is likely to encounter during normal operations. This section could also list the types of events an MTO should record in an operations log or run book. Because an online IMS operation is likely to span several operator shifts, some type of record of the MTO's actions is desirable.
- A problem-reporting procedure

  This section describes how to report problems or unusual events. An incident report is a useful tool to help you control the integrity of the system and service for the end user. Any of the following personnel or groups could fill in this report form: the end user or user liaison group, the IMS MTO or network administrator, the database administrator, or the system programmer.

Recommendation: Regardless of who fills in the form, you should require that it be completed, even though the person competing the form contacts other operating personnel. This form can become a controlling document in the case of a recovery action, but can also be used for minor complaints. To foster cooperation, you should return a copy of the report, showing actions taken, to the originator. You should also periodically review these reports to discover recurring problems and make valuable procedural changes.

Related Reading: See "Procedures for End-User Terminal Operators" on page 339 for considerations for remote terminal operation and end-user documents.

Setting the Level of Operator Control

For a complex system such as IMS, you need to carefully match operational control to operator capability. IMS is designed so processing is controlled by message traffic, which places control of the operation primarily in the hands of the master terminal operator. You must decide whether an operator or an application administrator should have primary control. From an administration point of view, control of operations is far more than a structured use of IMS commands.

The MTO can have a wide range of responsibilities. At one end of the spectrum, you have an operator using a standard set of instructions, designed using a cook-book approach: "if such-and-such happens, perform procedure X". At the other end of the spectrum, you have a system programmer who understands the flexibility of IMS control as well as the application program and database requirements. The first approach requires good symptom-based instructions; the second requires clear guidelines and periodic auditing.

If an operator handles your operations, the two areas for which he or she needs instructions, detailed to a level appropriate for his or her knowledge and responsibilities, are:

- Normal operations

  You need to document instructions for making specific resources available for committed end-user service. The instructions must be especially clear about when and how to shut down and restart IMS. If alternative network connections are part of the system design, the instructions should have a set of additional procedures for them.

  Operators who understand the end-user priorities should be allowed to change the scheduling algorithm by making reassignment of message priorities or even of message region classes.

- Error and recovery operations (triggered by symptoms or user feedback)

  These events range from temporary transaction processing problems to problems that have serious database implications. Your recovery procedures should include carefully documented and tested actions for your operators to take.

  Recommendation: Design your procedures so the operator initiates recovery actions in cooperation with your user-liaison group and your system-support personnel. Structure your operations to take advantage of as much consultation as possible. Insist that your operators record all recovery actions in a run log, including justification for decisions made.

  An important part of the operations strategy is the recovery procedures used to restart the online IMS subsystem after a failure of the operating system or the IMS control region.

Designing Record-Keeping Procedures

The MTO needs to keep various types of information that document system activity. Consider designing standard forms or online panels for recording this information. Following is a list of the general types of information the MTO needs to record:

- IMS control region activity
- Utility executions
- Application-program activity
- Problems or unusual events
- Startup, shutdown, and system log activity

Some of this information can be recorded automatically. DBRC records all OLDS and archive activity, and can also record the execution of batch jobs and recovery-related utilities.

The secondary master terminal provides a hardcopy log of IMS system activity, including checkpoint information and IMS messages. In addition, if you select the hardcopy logging option (in the system definition COMM macro), and you do not disable it using the `/SMCOPY` command, IMS logs certain IMS commands and responses to the secondary master terminal. This information can be helpful when tracking a series of events affecting the operation of IMS, or when verifying that an operations task was performed. Information useful for analyzing an IMS error situation is often in the MVS hardcopy log.

You can select whether IMS logs commands issued from the master terminal or commands entered from other terminals, or both. In either case, IMS only logs certain commands; Table 44 lists these commands. When you issue a command from a terminal other than the master terminal, IMS logs output and identifies its node, or line and physical terminal number. The `/SMCOPY` command starts and stops the hardcopy logging facility.

Related Reading: See *IMS Version 9: Command Reference* for details about this and other IMS commands.

Table 44. IMS Commands That Can Be Logged to the Secondary Master Terminal

/ACTIVATE	/DEQUEUE	QUERY	/RMNOTIFY
/ALLOCATE	/DISPLAY	/QUIESCE	/RSTART
/ASSIGN	/IDLE	/RCLSDST	/START
/CHECKPOINT	/MODIFY	/RCOMPT	/STOP
/CLSDST	/MONITOR	/RMCHANGE	/SWITCH
/COMPT	/MSASSIGN	/RMDELETE	/TRACE
/DBDUMP	/OPNDST	/RMGENJCL	/UNLOCK SYSTEM
/DBRECOVERY	/PSTOP	/RMINIT	UPDATE
/DELETE	/PURGE	/RMLIST	

Recording IMS Control Region Activity

You should design a form on which the MTO can document executions of the control region. Details to be recorded on the form include: the time when the system is started or shut down, the type of restart, and the reason for shutdown. You should make provisions on the form for recording error situations, such as DASD I/O errors, and actions taken to recover from such errors. The operator can file the output of the DBRC `LIST.LOG` command with this form to provide a record of log archive activity.

Recording Execution of IMS Utilities

How much manual record keeping is necessary for IMS utilities depends on your use of DBRC. If you have registered all your databases with DBRC, DBRC records the execution of the recovery-related utilities in the RECON data set. By listing the RECON data set (`LIST.RECON`) at regular intervals, you can easily maintain records of utility executions.

If you have not registered all your databases with DBRC, you should design forms for recording execution of the following utilities:

- Database Image Copy
- Database Image Copy 2
- Online Database Image Copy
- Database Change Accumulation
- Database Recovery
- Those utilities involved in database reorganizations

Figure 27 is a sample form for recording Database Change Accumulation activity. Figure 28 on page 316 is a sample form for recording Image Copy activity. These forms assume that changes for all databases are accumulated together. Figure 27 shows the input log data sets, and the output change accumulation data sets.

Run				Purge		Input Data Sets*	
Date	Time	Data Set Name	Volume Serial Number	Date	Time	Name	Volume Serial Number
91005	1520503	IMS.CUM	CUM03	91004	1200000	IMS.CUM	CUM02 SLDS01 RLDS01
91006	1510031	IMS.CUM	CUM04	91005			
*Includes old change accumulation data set, and SLDS or RLDS							

Figure 27. Sample Form: Running of the Database Change Accumulation Utility

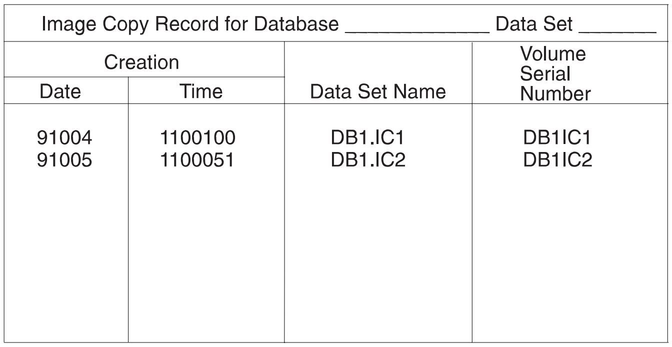


Image Copy Record for Database ____________ Data Set _______

Creation			Volume Serial Number
Date	Time	Data Set Name	
91004	1100100	DB1.IC1	DB1IC1
91005	1100051	DB1.IC2	DB1IC2

Figure 28. Sample Form: Running of the Image Copy Utility

These forms are only samples. Use them as templates to design forms that suit your installation's procedures. You also need to record the execution of batch database reorganizations. A database cannot be backed out to a point earlier than its latest batch reorganization. Nor can a forward recovery be done from an image copy preceding the latest batch reorganization. Thus, a knowledge of recent reorganizations is important when recovering a database.

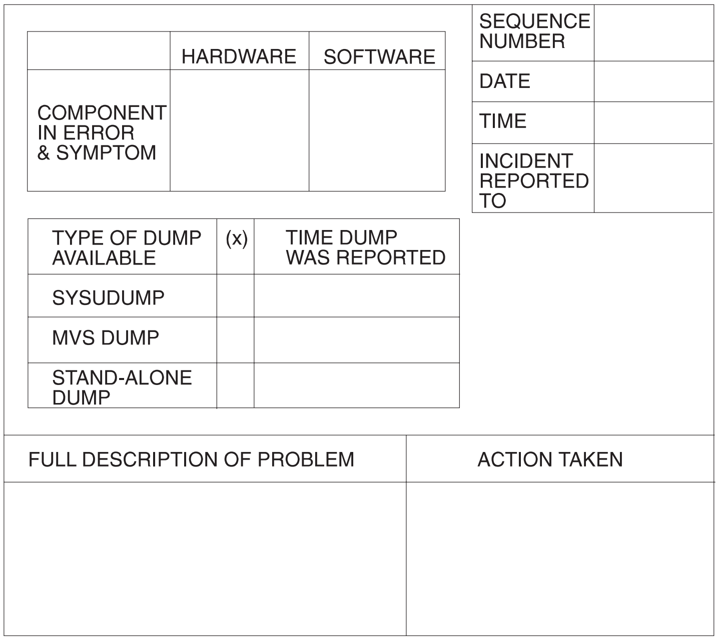
Recording Application-Program Activity

You might want to design forms to record the execution of BMPs and batch application programs. The latter is particularly important if you are not using DBRC to record logs produced by batch programs. Other information you might want to record includes checkpoint IDs and any error messages produced by the programs.

Recording Problems or Unusual Events

Another type of form you might want to design is one for recording any system problems or unusual events. Such a form is often called an incident report. Incident reports help you control the integrity of the system and services for end-users. This form might be filled in by the user liaison group, the MTO, or the recovery specialist.

Recommendation: Regardless of who fills in the form, you should require that it be completed, even though the person competing the form contacts other operating personnel. This form can become a controlling document in the case of a recovery action, but can also be used for minor complaints. To foster cooperation, you should return a copy of the report, showing actions taken, to the originator. You should also periodically review these reports to discover recurring problems and make valuable procedural changes. Figure 29 on page 317 shows a sample incident report form.



	HARDWARE	SOFTWARE
COMPONENT IN ERROR & SYMPTOM		

SEQUENCE NUMBER	
DATE	
TIME	
INCIDENT REPORTED TO	

TYPE OF DUMP AVAILABLE	(x)	TIME DUMP WAS REPORTED
SYSUDUMP		
MVS DUMP		
STAND-ALONE DUMP		

FULL DESCRIPTION OF PROBLEM	ACTION TAKEN

Figure 29. Sample Form: Incident Report

Recording Startup, Shutdown, and System Log Activity

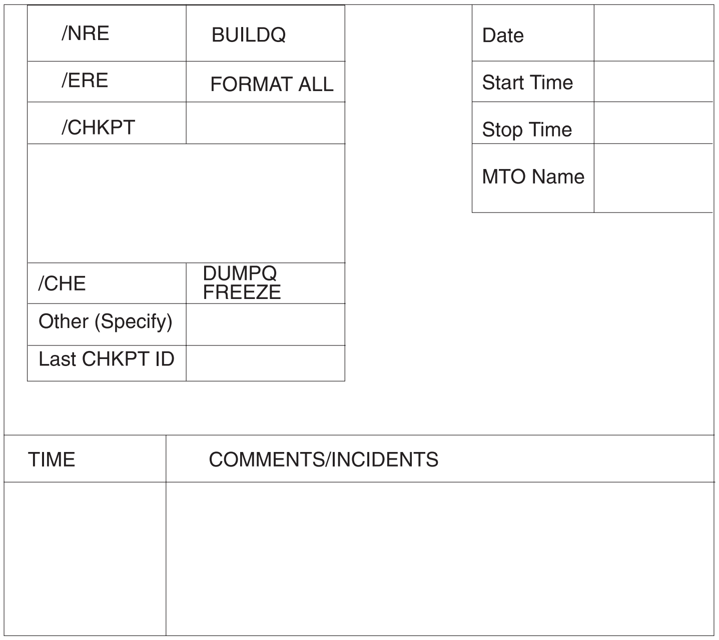
Another type of form you might want to design is one for recording startup, shutdown, and their relationship to system log activity. This form is often called an online run sheet. Figure 30 on page 318 shows a sample online run sheet.

The MTO fills in the online run sheet for each execution of the online system. When starting IMS, the MTO records (circles) the parameters used in the start command, along with the checkpoint ID used. When shutting down IMS normally, the MTO records (circles) the parameters used. When shutting down IMS abnormally, the MTO records the reason for the failure and fills in an incident report form. If the MTO shuts down IMS using a `/CHECKPOINT DUMPQ` command, he or she records the closing checkpoint ID from the `DFS994I` message.

The MTO should use the bottom half of the form (comments/incidents) to record any unusual incidents. The MTO should note such things as: hardware errors on telecommunication lines, details about reassignment of LTERM names, and abending transactions. Finally, the MTO should add a cross reference to the incident report, where necessary.

DBRC records information about use of OLDSs in the RECON data set. After IMS archives an OLDS for an execution of the online system, the MTO can issue a `LIST.LOG` command to get information about the use of the OLDS. The MTO should file the hardcopy listing of this information with the online run sheet. This information provides an important means of communication between one shift of MTOs and the next. This information can also be used to assist in reconstruction of the RECON data set, if it becomes damaged.

The MTO should also file the complete hardcopy listing from the master terminal printer with the online run sheet.



/NRE	BUILDQ
/ERE	FORMAT ALL
/CHKPT	
/CHE	DUMPQ FREEZE
Other (Specify)	
Last CHKPT ID	

Date	
Start Time	
Stop Time	
MTO Name	

TIME	COMMENTS/INCIDENTS

Figure 30. Sample Form: Online Run Sheet

Planning Availability of IMS Service

In order to provide a framework for the operator control of the online IMS subsystem, you need to obtain overall requirements. You can then proceed gradually to fill in the details of specific requirements of various user groups.

Defining the Production Cycle

The following are suggested steps you should take to define the production cycle:

1. Obtain the overall production cycle requirement.

   Example: Suppose the execution of an application program requires the online subsystem to be available from 0400 to 2000 hours, with no weekend requirements. Very often these requirements are made to fit the data processing organization’s constraints. In this example, the operations group must be agreeable to the early-morning scheduling (which is probably required for transaction input from locations in different time zones).

2. Obtain support requirements for databases and batch-program scheduling.

   Example: A daily batch-report program might cause certain databases to be unavailable until 0800. A `/DBRECOVERY DB` or `UPDATE DB STOP(ACCESS)` command stops additional transaction or CCTL access and allows offline processing for the database named in the command. Another example is a priority transaction, scheduled for database integrity purposes, that preempts other transaction processing.

Table 45 shows a checklist of initial availability requirements you need to collect.

Table 45. Availability Requirements Checklist

Availability Requirement	Follow-up Activity
What are the available hours: Daily? Weekly?	Check if a regular weekly time. Check daily start and stop times. Allow for weekends, holidays, and preventive maintenance.
Which are services critical to the user?	Summarize the transactions, programs, and database needs. Summarize the CCTL's dependencies for programs (PSBs) and databases.
What is the target maximum down time?	Establish guidelines for duration.
Are there any planned interruptions?	Check requirements for normal restart. Determine how queues are to be handled.
Are there variations of nucleus?	Match nucleus suffix to system definition schedule.
Is there any special message region scheduling?	Develop IMSMSG procedure variation. Find conditions that prompt scheduling.
What are the BMP scheduling requirements?	Tailor IMSBATCH procedure for BMP. Find MVS file requirements.
What databases are not always online?	List the databases, their conditions, and the reasons for their being offline.
What batch programs are to run during online schedule?	Find any database conflicts. Find the required database levels and restart instructions.
What is the timing of database reorganization and related jobs?	Find jobs that run during online execution. Obtain master schedule from DBA.
Do any application programs require exclusive processing?	Find the effect on other online processing.
Are there configuration variations or multiple IMS subsystems?	Develop procedure support: plan for MSC, data sharing, DL/I separate address space, DBCTL, DCCTL, APPC/IMS, OTMA, or ETO.
Are you using a data propagation manager?	Make sure both IMS and DB2 are available on the same MVS system. Make sure the program libraries containing the data propagation user exit routines are available to all dependent IMS regions.

Fixing Network Availability

The next level of availability concerns the network. Document which devices are to be initially available, those that should always be active, and those that have restricted usage. The important attribute is whether the MTO needs to activate the connection or must make any alternative connections in case of terminal or line failures.

A second aspect of availability is to ensure proper control of the network. You need to document what terminals are connected and when.

Table 46 identifies some connection data you need to collect.

Table 46. Checklist for Network Control Connection Data

Connection Data	Follow-up Activity
Which terminals are associated with services critical to the user?	List the connection requirements. List the LTERMs.
Are VTAM terminals to be continuously connected?	List those nodes and their LTERMs.

Table 46. Checklist for Network Control Connection Data (continued)

Connection Data	Follow-up Activity
Are VTAM terminals to be connected on demand?	Identify how each is controlled (MTO, VTAM operator, or remote terminal operator).
Are there planned reconfigurations?	Develop sets of terminals and identify how they should be controlled.
Are there alternate connection plans?	Identify if there are alternate pairings, or occasional usage.
Are intelligent stations as connected subsystems used?	Identify operations for connected subsystems, System/7, System/3, or Personal System/2, and monitoring requirements.
Are switched lines used?	Identify the users and give the LTERM names in the group.
Is there batched data entry or high-volume input?	Identify the pattern of batched devices.
Are there special output storage devices?	Identify use of printers or data devices.

Establishing Other Operations Responsibilities

A third checklist identifies the use of support services and expected MTO control actions. Table 47 presents this information; your emphasis should be on the operational requirements. It is important to understand why an application program places restrictions on what commands you can issue. For example, online image copy is required at end-of-shift, but, because of the requirements of an IMS restart, no conflict in database level is permitted. This kind of restriction might require you to prevent certain programs from executing.

Table 47. Checklist for Online Control Services

Check Requirement for:	Follow-up Activity
Dynamic allocation of databases	Requires coding of DFSMDA macro and prior cataloging of databases.
Online database image copy	List databases and schedule timing. Is image copy restart allowed?
Database surveyor	When to be scheduled. Who is the contact for range specification?
Database recovery control	Arrange for DASD dump entries. How are RECON data sets to be controlled? How are JCL generation facilities to be used?

Simple checkpoint
`/CHECKPOINT`
`PURGE`
`/CHECKPOINT DUMPQ`
`/CHECKPOINT FREEZE`
`/NRESTART`
	What conditions require the checkpoint? Is SNAPQ required?
	
	Must all queues be emptied?
	Can queues be held?
	What conditions warrant use of this command?
	Is queue rebuilding required?

Online system log management	Determine archive procedures and whether WADS and OLDS are to be dynamically allocated.
Online change management	Determine the scope and timing of online changes and how to control processing.
Offline dump formatting management	Establish transfer, archive, and formatting procedures.
IBM 3850 Mass Storage System (MSS)	Obtain scenario of IBM 3850 usage.

Table 47. Checklist for Online Control Services (continued)

Check Requirement for:	Follow-up Activity
Monitoring	How will IMSMON data set be handled? What requirements exist for traces or statistics?
Security Signon Transaction Refresh matrix	Any override conditions? Any override conditions? What is update frequency and control procedure?

Controlling Operations with the MTO

When you are defining the duties of the master terminal operator, you must decide what actions are part of the normal duties, and what decisions require consultation with other groups.

From the end-user viewpoint, the MTO could represent the data-processing staff, as well as the application-program contact. You need to arrange ahead of time what kind of support can be provided by the MTO, and if there is any way of shielding the MTO from unnecessary interruption. The MTO's primary duty is to monitor the condition of the total IMS online environment, and to maintain normal service. Any alterations in committed service or recovery decisions are really part of a wider set of business responsibilities, not necessarily those of the MTO.

The master terminal operator has the following abilities and responsibilities:

- Responsibility for running IMS

  The MTO starts and shuts down the IMS subsystem, starts regions, and manages the system log.
- Knowledge of the ongoing status of the IMS subsystem

  The MTO continuously monitors processing and the detects any error situations.
- Control over contents of the system and the network

  The MTO can control the network, connect other IMS systems, and perform other prearranged tasks.
- Privileged commands

  In addition to the routine work, the MTO responds to error conditions, changes the scheduling algorithm, alters passwords, and reconfigures the system as necessary.

Setting Up for Normal Operator Actions (DB/DC or DCCTL)

The MVS system operator needs to initialize the IMS control region, but you should plan to reduce the need for further involvement by the MVS operator in IMS operations. In a busy system, responses to messages or instructions sent by an application program to the MVS operator can be erratic. Such requirements are better handled by output messages to the IMS master terminal.

You should ask the MVS system operator to relay any IMS messages (ones beginning with any of the following prefixes: DFS, DSP, DXR, ELX, BPE, or CQS) to the IMS master terminal. You might also give the MVS operator a list of the IMS job names. Because often the MVS system console is physically separated from the IMS master terminal, you need a local telephone or a paging line to maintain communication between them.

You need to decide whether the MVS operator or the IMS MTO should start IMS regions. If you want the MVS operator to start them, he or she can use one of the following IMS procedures:

- IMSBATCH procedure to start batch message processing regions
- IMSRDR procedure to start message processing regions

The advantage of allowing the MVS operator to start IMS regions is that he or she can monitor any abnormal termination patterns and I/O requests.

Table 48 shows the actions usually performed by the MTO and the commands usually reserved for the IMS master terminal.

Table 48. Master Terminal Operator Actions Usually Performed

Activity	IMS Command
Activate IMS (cold start)	/ERESTART COLDSYS
Start a message region	/START REGION IMSMSG1
Start communication lines	/START LINE ALL
Display message queues	/DISPLAY
Start another message region	/START REGION IMSMSG3
Prepare for VTAM communication	/START DC
Initiate static VTAM sessions	/OPNDST NODE ALL
Initiate dynamic VTAM sessions	/OPNDST NODE AOI
Send message to terminals	/BROADCAST
Shut down VTAM terminals and IMS	/CHECKPOINT FREEZE QUIESCE
Restart IMS (warm start)	/NRESTART

The IMS master terminal also works well as a display device. On the bottom of the screen are two input lines, with room for a warning message. The remainder of the screen is divided into two areas, each of 10 lines. The top portion shows the 10 most recent IMS messages; new messages overlay old ones, so the display cycles through the 10 lines. The lower portion is used to page through the output from the `/DISPLAY` command.

If you set the master terminal to use a 3270 format, some of the program function keys are predefined as follows:

PF Key	Command or Function
1	`/DISPLAY`
2	`/DISPLAY ACTIVE`
3	`/DISPLAY STATUS` or `QUERY STATUS`
4	`/START LINE`
5	`/STOP LINE`
6	`/DISPLAY POOL`
7	`/BROADCAST LTERM ALL`
11	Next message page (NEXTMSGP)
12	Copy function

Recommendation: Create a PF key panel that shows these settings and place it with the keyboard as a reminder for the operator.

Setting Up for Normal Operator Actions (DBCTL)

The DBCTL MTO can control the DBCTL environment from any MVS system console. If your MTO operates from a different console (secondary system console) from the primary one where the MVS operator sits, you must ensure a means of communication between the two operators. DBCTL requests for tape or DASD mounts must be given to the system console operator. Some IMS messages (DFS prefix) originate from the CCTL region and thus appear on the primary system console; these messages should be relayed to the MTO in case some action needs to be taken. For example, when the CCTL tries to connect to a DBCTL environment and the DBCTL subsystem is not yet running, the IMS Database Resource Adapter (part of the CCTL region) issues message `DFS690A`. In general, communication between the CCTL operator and the MTO is important.

As in the DB/DC or DCCTL environments, the MVS operator can start IMS BMPs by using either the IMSRDR or the IMSBATCH procedure. Any activity of IMS dependent regions or utilities the MVS operator controls should be recorded or communicated so that the MTO is aware of all IMS-related activity.

Table 49 shows the actions usually performed by the MTO and the commands usually reserved for the IMS DBCTL master terminal.

Table 49. Master Terminal Operator Actions Usually Performed

Activity	IMS Command
Activate IMS (cold start)	/ERESTART COLDSYS
Start a BMP	/START REGION IMSBMP
Display dependent regions	/DISPLAY ACTIVE
Shut down IMS	/CHECKPOINT FREEZE
Restart IMS (warm start)	/NRESTART

Specifying Resources Controlled by the MTO

Table 50 summarizes the resources controlled by the MTO.

Table 50. Resources Controlled by the MTO

Communications	Scheduling	Integrity
Telecommunication lines	Regions	Checkpoint
Physical terminals	Databases	Restart
Logical terminals	Transactions	Database copy
Spooled Output	Programs	Database availability
Users		System status

Planning for Network Operations

The major part of any detailed instructions for the MTO is the required connections or any alternate connections (in the case of terminal or line failures). A useful aid for the MTO is a chart showing the following:

- The network
- The device type
- PTERM and LTERM names

- Availability requirements
- A summary of the application usage

The chart should specify the contacts for problem resolution, whether they are user liaison, VTAM operator, or some other form of network control.

Session Initiation and Termination Options

Because of their functional capability, many VTAM-supported terminal devices are shared by IMS and other users. You have several choices for handling connect and disconnect. You can have the remote operator or master terminal operator be responsible, or you can call upon intervention by a VTAM operator. Table 51 on page 324 summarizes your choices.

Table 51. Session Initiation and Termination Options

Control by	Initiate by	Terminate by
MTO	/OPNDST	/CLSDST
Remote operator	LOGON IMS	/RCLSDST
(VTAM,Defn)	Initialization list	-
VTAM operator	VARY command	VARY command

Auditing Operational Control

To understand the actions taken by the MTO, you should obtain a permanent printed copy of the command sequences issued by the MTO. This function is termed hardcopy logging. Copies of an entered command and its response are sent to the terminal designated as `SECONDARY` in the system definition NAME macro.

Some IMS commands are automatically copied to the secondary master terminal. If you specify `COPYLOG=ALL` in the COMM macro during system definition, then when other terminals besides the master terminal enter any of these commands, their command use is also recorded on the secondary master terminal. The MTO can use the `/SMCOPY MASTER OFF` command to override the logging of commands to the secondary master terminal. The MTO can also use this command to turn off logging of command entry and responses from other terminals. To resume logging to the secondary master terminal, the MTO can use the `/SMCOPY MASTER ON [TERMINAL ON]` command.

This more permanent record is valuable for several reason, but the principal one is that of accountability. You can also use the output to audit whether the operations command sequence and timing are providing committed service. The checklist shown in Table 52 on page 325 identifies the types of activities and related commands, and shows some of the ways you can use the secondary master terminal output.

Table 52. Using Secondary Master Terminal Output to Audit Performance

Type of Activity	Command	Output Usage
Network control	/OPNDST	Check VTAM connection procedure.
	/CLSDST	Check VTAM disconnect procedure.
	/RCLSDST	Check expected disconnect.
	/START	Check network availability.
	/RSTART	Coordinate with terminal usage.
	/STOP	Investigate stopped resource.
	/IDLE	Investigate if there is lost or waiting output.
	/MSASSIGN	Check if planned MSC connect.
Device control	/COMPT	Check planned VTAM component switch.
	/RCOMPT	Check remote VTAM component allowed.
	/MONITOR	Monitor System/3, System/7 operation.
Resource control	/ASSIGN	Check that all uses fit operations plan.
	/START	Check region availability and queue control.
	/STOP	Investigate all events.
	/PSTOP	Investigate why input stopped.
	/PURGE	Investigate why service was restricted.
	/MODIFY	Audit the extent and timing of online changes.
	/CHANGE	Check size of MFS dynamic directory.
Security	/DELETE	Audit reason for password or terminal security deletion.
	/MODIFY	Set timing of authorization and password changes.
Recovery	/DBRECOVERY	Investigate database errors.
	/DBDUMP	Investigate database errors and subsequent processing.
	/DEQUEUE	Investigate lost message or output.
	/CHECKPOINT	Coordinate with reported errors.
	/SWITCH	Investigate reasons for unplanned takeover activity.
	/UNLOCK SYSTEM	Investigate takeover activity.
Monitoring	/DISPLAY or QUERY	Incorporate in monitoring feedback. Check use for scheduling.
	/TRACE	Coordinate with trace output.
HALDB OLR	/INITIATE	Initiate or resume HALDB Online Reorganization (OLR).
	/TERMINATE	Stop HALDB OLR.
	/UPDATE	Change the rate or the disposition flags of the DEL or NODEL data sets of a HALDB OLR.

Operator Control of Conversational Transactions

The master terminal operator should be aware of the potential use of conversational transactions because their queue characteristics are somewhat different from those of nonconversational transactions. The number of conversational transactions that can be active at one time is 65535 per user or terminal. Also, the same individual transaction can be scheduled many times, but the processing patterns can be quite different among the set. For example, one conversational transaction could involve an extended search of a database, while another, using multiple-choice menus, might retrieve a single item of data. Another consideration for conversational transactions is whether the conversations are to be active until they are complete or whether they can be held while the end user is engaged in an offline activity.

Controlling Conversational Transactions

Table 53 summarizes the type of information that influences the content of your operational procedures when the online IMS subsystem includes conversational transactions.

Table 53. Operator Information for Conversational Transactions

Transaction Attribute	Information Required	Procedure Impact
Which transactions are conversational?	Summary of purpose and processing characteristics.	Multiple messages might need the message format buffer pool and message queues to be monitored.
Any dedicated terminals? What is the maximum number allowed to run?	Name LTERM and PTERM. Find maximum storage using DASD and expected loads.	Know impact of terminal problem. Advise remote terminal operator of restrictions and status.
Can conversations be held or released?	If so, how long? Is operator allowed to terminate?	MTO awareness of status. Use of `/PSTOP` or `UPDATE` and `/EXIT` commands. Use of `/MODIFY` warnings.
What recovery actions are to be taken?	At restart. Are abnormal termination exit routines used?	Notify remote terminal operator and other recovery personnel of output loss.

Remote Terminal Operator Control for Conversational Transactions

A remote terminal operator can use a `/FORMAT` command to begin a conversation. This command formats an MFS-supported terminal screen. Or, the terminal operator can enter the initial transaction. To terminate the conversation before the normal program end, the terminal operator can use the `/EXIT` command.

The operator can suspend the conversation (if he or she needs to leave the terminal area) by using the `/HOLD` command. In response, IMS issues message `DFS999I`, which displays the identification number needed to resume the conversation with the transaction program. The terminal is then available for other transaction activity.

On return to the terminal, the terminal operator can issue the `/RELEASE CONVERSATION` *nnnn* command to resume the conversation, where *nnnn* is the conversation ID from the `DFS999I` message. As a convenience, IMS re-sends the last output message to the terminal. Or the operator can issue the `/EXIT CONVERSATION` *nnnn* command to terminate the conversation.

If the MFS formatting or the output content of the screen is lost during a conversation, enter the `/HOLD` command followed by the `/RELEASE` command. These commands resend the first physical page of the current message and restore the format.

MTO Control for Conversational Transactions

The master terminal operator can obtain a status of all conversations, held or active. The following output from the `/DISPLAY CONVERSATION` command displays the terminal address, the conversation identification number, and status and type of all conversations:

```
TERMINAL USER    ID STATUS
11-   2          0001 HELD
 4-   2          0002 ACTIVE,SCHEDULED
 4-   1          0011 HELD
VT03             0012 ACTIVE
```

The MTO can terminate a conversation. For example, the `/EXIT CONVERSATION 0001 LINE 11 PTERM 2` command terminates the first conversation shown in the command output above. A `/START LINE`, `/START NODE`, or `/START USER` command also terminates the conversation, unless a program is already scheduled to process a portion of the conversation exchange.

Recommendation: Before terminating a conversation, the MTO should broadcast a warning message to that terminal.

Allowing for Conversation Abnormal-Termination Exit Routines

When a conversation terminates abnormally, a termination exit routine named DFSCONE0 can gain control. Its use might be to initiate cleanup activity, or simply to notify the input terminal of the canceled conversation. If the exit routine determines that more elaborate correction is required, it can schedule a transaction. This transaction can include the latest SPA, or a modified version of it, as a message segment. IMS provides a default exit routine (DFSCONE0) which schedules a transaction that includes the old SPA.

You can use the conversational termination exit routine to determine the effect of operator interruption.

Related Reading: For more information about the uses of these routines and the “clean-up” transactions, see *IMS Version 9: Customization Guide*.

MTO and Operations

The MTO or an authorized remote terminal operator can initiate several kinds of trace activity with the `/TRACE` or `UPDATE` command. Because trace or monitoring actions affect the performance of the online IMS system, they are not part of normal operations. Even though you might use the IMS Monitor fairly regularly, you should regard all tracing operations as “special requests”. The validity of the trace depends upon the processing taking place. Also, the operator can control the trace interval with the `/TRACE` or `UPDATE` command, so the operator must be aware of the trace timing.

Some types of trace requests do not require the person requesting the trace to be present during normal operations. The request should state the approximate duration of the trace and the type of monitoring. The main section should specify the initial start and termination conditions, as well as any monitoring constraints to be applied. Other sections should prompt for other details, such as queue levels, transactions to be active, or the time of day.

The types of tracing that can be requested are summarized in Table 54 on page 328.

Table 54. Trace and Monitor Options

Type of Trace	/TRACE Parameters	Operational Effect
IMS Monitor	SET MON opt	Initiates IMS Monitor trace.
	SET OFF MON	Terminates the trace.
	opt=ALL	Traces all events.
	opt=APDB	Traces activity between application programs and databases.
	opt=APMQ	Traces activity between application programs and message queues.
	opt=LA	Traces line and logical link activity.
	opt=SCHD	Traces scheduling and termination events.
DL/I Image Capture	SET PSB name opt	Traces all DL/I calls for the named PSB.
	SET OFF PSB name	Terminates the trace.
	opt=COMP	Generates PCB and data-compare statement images.
	opt=NOCOMP	Generates no compare statement images. Default.
	SET PI OPTION opt	Traces program isolation activity, including ENQ, DEQ, and DL/I calls. Trace data is written to system log.
	SET PI	Traces PI activity. Trace data is kept in storage, and not written to system log.
	SET OFF PI	Terminates the trace.
	opt=LOG	Writes trace data to system log. Default.
	SET OFF PI OPTION LOG	Continues trace, but does not write buffers to system log.
	opt=TIME	Adds elapsed time for each ENQ and DEQ call when IMS has to wait.
	SET OFF PI OPTION TIME	Continues trace, but only records time of day.
	opt=ALL	Equivalent to opt=LOG TIME.

Table 54. Trace and Monitor Options (continued)

Type of Trace	/TRACE Parameters	Operational Effect
Line, node, and PTERM tracing	SET LINE nn opt	Traces line activity.
	SET OFF LINE nn	Terminates the line trace.
	SET NODE nn opt	Traces node activity.
	SET OFF NODE nn	Terminates the node trace.
	opt=LEVEL n	Sets detail level (1-4) for trace.[1]
	opt=MODULE mod	Traces modules for IMS communications manager.
	mod=DDM	Traces device-dependent modules.
	mod=MFS	Traces MFS modules.
	mod=ALL	Traces both DDM and MFS modules.
Internal Trace Tables	SET TABLE opt OPTION LOG	Initiates tracing into specified trace tables. IMS writes trace tables to log before reusing them.
	SET TABLE opt OPTION NOLOG	Initiates tracing into specified trace tables. IMS does not write trace tables to log. Default.
	SET OFF TABLE Opt	Terminates trace.
	opt=ALL	Specifies that IMS enable or disable all traces. Default.
	opt=DISP	Traces dispatching events.
	opt=DL/I	Traces DL/I calls.
	opt=DLOG	Traces log activity.
	opt=FAST	Traces Fast Path activity.
	opt=IDC0	Traces errors in modules DFSCNXA0 and DFSIDC00.
	opt=LATC	Traces latching activity.
	opt=LOCK	Traces locking and PI activity.
	opt=LRTT	Traces log router activity.
	opt=LUMI	Traces LU 6.2 and APPC activity.
	opt=OTMT	Traces OTMA activity.
	opt=RETR	Traces DL/I retrieve calls.
	opt=SCHD	Traces scheduling events.
	opt=STRG	Traces storage manager activity.
	opt=SUBS	Traces external subsystem events.

Table 54. Trace and Monitor Options (continued)

Type of Trace	/TRACE Parameters	Operational Effect
Link Tracing for MSC	SET LINK nn opt	Traces logical link.
	SET OFF LINK nn	Terminates link trace.
	opt=LEVEL n	Sets detail level (1-4) for trace.[1]
	opt=MODULE mod	Traces modules for IMS communications manager.
	mod=DDM	Traces device-dependent modules.
	mod=MFS	Traces MFS modules.
	mod=ALL	Traces both DDM and MFS modules.

Note:

[1] The detail levels for control blocks are described in *IMS Version 9: Command Reference*, under the /TRACE command.

IMS can trace several different types of activity concurrently. The IMS Monitor writes its output to a data set specified on the IMSMON DD statement or dynamically allocated. IMS writes other traces to type X'67' log records.

Offline Dump Formatting Operations

IMS operators need to be aware that their intervention might be required if a dump is requested at a time when the dump data sets are unavailable or when some other task in the MVS system is currently taking an SDUMP. In these situations, IMS issues message DFS3906 or DFS3907. In addition, CCTL operators must be aware that the CCTL can issue these messages when the IMS database resource adapter (DRA) is trying to generate an SDUMP.

Related Reading: For more information about these messages, see *IMS Version 9: Messages and Codes, Volume 1*.

Designing Operating Procedures

In general, you can document operating procedures in one of two ways:

- Using flowcharts or a similar graphic device
- Using a narrative style

Related Reading: The types of procedures you need to develop for operating IMS are discussed in detail in Part 2, “Operating an IMS Subsystem,” on page 21.

Using Flowcharts and Other Graphic Techniques

When you use flowcharts or other graphics to document operating procedures, you generally accompany them with an expanded textual description of the various steps in the flowchart.

Flowcharts have several obvious advantages over narrative presentations of procedures. For example, with a flowchart it is easy to get an overall picture of what needs to be done. Another advantage is that when text accompanies each flowchart, the reader can skip reading any text associated with a step he or she already understands. If the flowchart, for example, says to start IMS and this step is understood, the reader does not need to read that “the system operator enters the /NRE command from the system console...” and so forth.

The main disadvantage to using flowcharts is that they are often difficult to maintain. Small changes to an operating procedure could require you to redesign an entire flowchart.

You might not need to use flowcharts for the IMS MTO. However, it is probably the preferred technique when designing procedures for a recovery specialist or others who need an overview of how something works or what actions are to be performed.

Using Narrative

There are many ways to document procedures using a narrative style, rather than graphics. For example, you could use a numbered list, where the reader starts at number 1, compares the situation with the information in number 1, then continues to number 2 or maybe jumps to number 15 depending on the decision made in number 1. This technique is essentially a narrative flowchart. As another example, you could write scenarios, wherein the reader picks a scenario that matches their situation, and follows all the steps in the scenario.

The advantage to this narrative style is that it is relatively easy to maintain. There are no graphics to redraw, and changes to the procedures seldom require much redesign. The disadvantage is that the overall operation being performed is not always obvious. In addition, if the user of the procedure suspects that he or she has taken a wrong branch in the procedure, it is difficult to backtrack through the steps.

Establishing Operating Procedures for Multiple Systems

In a non-sysplex environment, each system in a multisystem configuration is operationally an independent unit. It exclusively owns its own communication resources, and is controlled by its own master terminal.

Multisystem Communication Initialization

Communication between two IMS systems using non-VTAM connections does not begin until the both MTOs issue an /RSTART LINK command (one for each system). When VTAM controls the physical link type, only one of MTOs needs to issue the /RSTART LINK command. Normally, the MTO would issue this command during IMS startup. VTAM only allows communication if the characteristics of the specified links are compatible. If a required link does not successfully start, IMS messages wait until the links are reassigned.

Attention: If you cold start a system that has messages queued, these messages are lost. Because the messages that were lost could be from or destined for terminals and programs in other systems, the impact of a cold start is not limited to the cold-started system.

Multisystem Communication Termination

A /PSTOP LINK command from either of two linked systems terminates transmission on the specified link. When transmission terminates on one side, its partner in the other system terminates its own transmission and notifies the MTO.

Use the command forms shown in Table 55 on page 332 to terminate physical links between partner MSC systems.

Table 55. Commands to Terminate Physical Links

Command	Keyword	MSC Use and Effect
/PSTOP	LINK	IMS halts transmissions associated with the logical link between two partner systems, but continues queuing for the remote resources. IMS enqueues, but does not send, broadcast messages that would use the link. IMS stops the logical link in the partner system, and sends message DFS2161 to its MTO.
	PURGE	IMS forces PSTOP condition even if other system failed while channel-to-channel (CTC) I/O was in progress. CTC link type only.
/IDLE	LINK	IMS forces termination of all transmissions on the physical link associated with the named logical link. Use only after a shutdown checkpoint.
	NOSHUT	IMS forces the IDLE condition without taking a shutdown checkpoint. Synchronous data link control (SDLC) link type only.

IMS Commands That Affect MSC Operation

This section contains information about the use of IMS commands that have a particular effect relative to the use of IMS Multiple Systems Coupling (MSC). See *IMS Version 9: Command Reference* for the definition and format of these commands.

Logical Link Assignments

Normally, you define initial logical link assignments (logical link to physical link) during IMS system definition. You can use the /MSASSIGN command to make or change a logical link assignment dynamically.

Recommendation: Use the /MSASSIGN command to create logical links only for unscheduled reassignments resulting from failing physical connections or systems.

Because a logical link must **always** communicate with its partner, the MTOs for the two systems must coordinate their assignments of corresponding physical links. You can replace any type of physical link with any other type of physical link.

Changes to logical link assignments remain in effect until you change them using an /MSASSIGN command or cold start IMS. The command only alters the relationship of MSC resources in the local system. The relationships that you can alter include those shown in Table 56.

Table 56. Logical Link Relationships That You Can Alter

Logical Link (Rel Link No.)	to	Physical Link (MSPLINK Name)
Remote Resource (SYSID)	to	Logical Link (Rel Link No.)
Logical Link Path (MSNAME Name)	to	Logical Link (Rel Link No.)

You can also use the /MSASSIGN command to do the following:

- Change a remote program to local
- Change a local program to remote
- Assign a logical path

When using the /MSASSIGN command, the following rules apply:

- You must assign operating logical links on a one-for-one basis to physical links, except for VTAM-type links with parallel sessions in effect. In this case, you can assign multiple logical links to one physical link.
- Before reassigning logical links, you must halt them using the /PSTOP command and they must be idle.
- You cannot reassign a destination SYSID to a logical link unless its currently assigned logical link is idle (after a /PSTOP command).
- MSC communication can occur only when logical links in two IMS subsystems share the same partner ID and are assigned to an operating physical link.
- The /MSASSIGN command does not determine if the requested assignment is reasonable or results in a valid configuration for MSC communication. IMS can only make this determination by communicating with remote system. You can accomplish this task by using the /MSVERIFY command after making assignment changes.

If a restart is pending on a logical link because of physical link failure, both systems should use the following procedure to reestablish communication through an alternate physical link:

1. Reassign the logical link to the alternate physical link.
2. Use the /RSTART LINK command to start the logical link.

If the link uses VTAM, you must issue the /PSTOP MSPLINK command before assignment either to or from the VTAM MSPLINK name.

Commands That Help Control Resources

You can use a number of commands to control resources in an MSC environment; see Table 57.

Table 57. MSC Environment Commands

Command	MSC Use and Effect
/ASSIGN or UPDATE	Change transaction priorities.
/CHANGE	Change link mode table and session restart.
/DELETE	Remove password security.
/PSTOP or UPDATE	Enqueue for remote processing, but do not send.
/PURGE or UPDATE	Reject further primary requests for remote processing. Accept secondary or continuing conversational requests. Continue sending those requests that are allowed to enqueue.
/STOP or UPDATE	For remote program or terminal: Return message DFS065 to input terminal for primary requests. Enqueue but do not send secondary requests. For local program or terminal accessed by remote MSC systems: Enqueue secondary requests. Return message DFS065 to input terminal (in input system) for primary requests. This message identifies, by SYSID, the MSC processing system that rejected the message. IMS logs the message, but cancels it so it is not processed.
/START or UPDATE	Reset previous effects of /START, /PSTOP, and UPDATE commands.

Table 57. MSC Environment Commands (continued)

Command	MSC Use and Effect
/BROADCAST	For local system only: Sends the broadcast message to all terminals when you specify `LTERM ALL`. There is no general broadcast capability to remote MSC terminals. You can use the keyword `MASTER` to broadcast to MTOs of remote MSC systems, even when the MTO terminals are not defined to the broadcasting system. However, you must define the SYSIDs of such remote systems using the MSNAME macro.
/TRACE	Trace MSC events using the `LINK` keyword. IMS records trace data in type X'67' records on the IMS system log.
Note: See Table 66 on page 381 for the appropriate keywords for these commands.	

If an MSC error occurs when the MSC trace is not operational, check the IMS log for X'67' records because IMS logs some error information even if the link trace is not on.

Use the following form of the `/TRACE` command when you suspect link problems:

```
/TRACE SET ON LINK linkname LEVEL 3 MODULE DDM
```

Managing the MSC Environment with the /DISPLAY Command

You can use the IMS display command to help manage the MSC environment. The command only displays information for the local system, however, and does not provide information available only from remote systems. Table 58 summarizes the information that is available to the MTO using the `/DISPLAY` command.

Table 58. MSC Information from the /DISPLAY Command

Command	Keyword	MSC Use and Effect
/DISPLAY	ASMT	Display current assignment of MSNAME to logical link and logical link to physical link.
	MSPLINK	Display the following: • Logical link (#) • Physical link (MSPLINK name) • Physical link type (MTM, BSC, VTAM, CTC) • Physical link address • Maximum parallel session number for SDLC link type
	LINK	Display the following: • Logical link (#) • Physical link (MSPLINK name) • Local and remote SYSIDs by MSNAME • Resynchronization for SDLC link type (forced or not)
	SYSID	Display the same information as for the `/DISPLAY ASMT LINK` command.
	MSNAME	Display the same information as for the `/DISPLAY ASMT LINK` command.

Table 58. MSC Information from the /DISPLAY Command (continued)

Command	Keyword	MSC Use and Effect
/DISPLAY	LINK *nn*	Display the following: • Logical link (#) • Partner ID • Number of messages received, sent, enqueued, dequeued, currently queued for link, link status • Mode table names and ASR status
/DISPLAY	MSNAME *nn*	Display the following: • Total messages enqueued for MSNAME (Logical Link path) • Total messages dequeued for MSNAME • Messages currently queued for MSNAME

Logical Link Path Control

You can use several commands to control logical link paths. Logical link paths are defined by SYSID pairs that identify the sending and destination systems. You define logical link paths using the MSNAME system definition macro. The logical link path is the lowest level of control across an MSC environment because it is the lowest level that is necessarily defined in intermediate MSC systems. Table 59 summarizes the commands you can use for link control.

Table 59. Commands Used to Control MSC Link Paths

Command	Keyword	MSC Use and Effect
/START	MSNAME	Start a previously stopped MSNAME.
/STOP	MSNAME	Stop the sending or receiving of primary request messages associated with the logical link path. When stopped by an input system, primary requests for remote programs or terminals associated with the stopped logical link path are canceled and IMS returns message DFS065 to the input terminal. Conversations in progress are allowed to continue. When stopped by a destination system, messages received from other systems over a stopped logical link path cause a logical link path to be stopped in the sending system (input or intermediate); IMS issues messages DFS2140 and DFS2142 (respectively) to the MTOs of the sending and receiving systems. Messages remain enqueued in the sending system until the logical link is subsequently started in both systems.
/PURGE	MSNAME	Halt enqueuing of primary requests for all remote terminals and programs represented by the MSNAME. Continuing conversations and secondary requests are still handled. Primary requests entered through an input terminal receive message DFS065. Requests from other systems that require use of the logical link path for a response are not accepted, but remain enqueued in the sending system (see /STOP MSNAME above).

Recovery Considerations for Multiple Systems

Each system in the multisystem configuration uses the full recovery capabilities of IMS. These capabilities assure that messages are not lost or duplicated within the single system if:

- You do not cold start the subsystem or emergency restart (BUILDQ) it using an earlier checkpoint.
- No log records are lost.

VTAM provides message integrity for SDLC links in addition to the MSC control functions.

Message Recovery

IMS assures that messages are not lost or duplicated across an MSC link by logging information about a message in both the sending and receiving systems. IMS restores this information during restart and exchanges it between the systems after the link is established. The sending system can then dequeue a message that was received by the receiving system but for which the acknowledgment was lost due to a link or system failure. The sending system can also re-send a message that was sent but not enqueued by the receiving system. If an IMS subsystem fails to recover, the messages for which it has recovery responsibility are lost.

Because IMS provides commands to dynamically change link assignments, you can set up an alternate processor for an inoperable one. The IMS system that resides in the inoperable processor can run in the alternate processor after all involved links are properly reassigned by the MTOs.

Stopped Transactions

If IMS stops a destination transaction, the action taken by the destination system varies based on the type of request:

- For a primary request that is not conversational or that starts a conversation, IMS sends an error message to the input terminal and cancels the message.
- For a primary request that continues a conversation or a secondary request, IMS enqueues the message. If the request is the first one received for that stopped transaction, IMS also sends a message to the MTO at that transaction's local system.

Application Program Abnormal Termination

When an application program abnormally terminates, and the abnormal termination is not the result of a deadlock situation, IMS issues message DFS554 to the master terminal of the system in which the abnormal termination occurred. This message includes the logical terminal name of the input terminal. If the input message is still available, IMS issues message DFS555, which includes the first portion of the input message, to the input terminal. When IMS sends the DFS554 message, the message text includes the logical terminal name of the input terminal.

Establishing Maintenance Procedures

The major task of maintenance is one of running IMS utilities on a regular, prescribed basis. The frequency with which utilities are run is an installation-dependent decision.

Related Reading: The IMS utilities references, *IMS Version 9: Utilities Reference: System* and *IMS Version 9: Utilities Reference: Database and Transaction Manager*, offer some guidance on how often to run the recovery utilities.

Table 60 lists the utilities you might want to run on a regular basis.

Table 60. IMS Utilities Run for Maintenance

Utilities	Utilities (continued)
HISAM Reorganization Unload (DFSURUL0)	File Select and Formatting Print (DFSERA10)

Table 60. IMS Utilities Run for Maintenance (continued)

Utilities	Utilities (continued)
HISAM Reorganization Reload (DFSURRL0)	Log Transaction Analysis (DFSILTA0)
HD Reorganization Unload (DFSURGU0)	Fast Path Log Analysis (DBFULTA0)
HD Reorganization Reload (DFSURGL0)	DB Monitor Report Print (DFSUTR30)
DB Surveyor (DFSPRSUR)	IMS Monitor Report Print (DFSUTR20)
Partial DB Reorganization (DFSPRCT1)	Spool SYSOUT Print (DFSUERT0)
DB Prereorganization (DFSURPR0)	MSDB Maintenance (DBFDBMA0)
DB Scan (DFSURGS0)	MSDB Dump Recovery (DBFDBDR0)
DB Prefix Resolution (DFSURG10)	DEDB Sequential Dependent Scan (DBFUMSC0)
DB Prefix Update (DFSURGP0)	DEDB Sequential Dependent Delete (DBFUMDL0)
DB Image Copy (DFSUDMP0)	DEDB Direct Reorganization (DBFUMDP0)
Online DB Image Copy (DFSUICP0)	Database Recovery Control (DSPURX00) [1]
DB Change Accumulation (DFSUCUM0)	
Log Archive (DFSUARC0)	
IMS Statistical Analysis (DFSISTxx)	

Setting Up Standard JCL

Another part of establishing procedures for your operations people is setting up standard JCL for the jobs that must be run (such as the utility jobs). Using DBRC can greatly assist you in this area, even though having completely “canned” JCL is not possible. For example, the MTO must supply the required log data sets when running the Database Batch Backout utility. If you use DBRC for recovery control, you may only have to set up JCL for DBRC itself.

Restriction: DBRC does not develop JCL for backout.

Testing Your Procedures

After you have prepared procedures for operating IMS, all those who operate the system should be given an opportunity to familiarize themselves with the procedures. After that, all procedures should be thoroughly tested.

Simulating System Failures

To test some of the recovery procedures, certain types of system failures must be simulated. Table 61 shows a table of possible failures and how they can be simulated.

Table 61. Simulating System Failures

System Failure	Simulated by
BMP or CCTL thread abended or canceled in error	Use MTO command `/STOP REGION n ABDUMP`.
IMS control region abended or canceled in error	Use MVS `MODIFY` or `CANCEL` command.

[1] Use this DBRC utility to delete log entries, using the `DELETE.LOG INACTIVE` command.

Table 61. Simulating System Failures (continued)

System Failure	Simulated by
MPP abended, looping, or in wait state	Use one of your own programs. Or use the DL/I test program DFSDDLT0; use the special calls ABEND and ZING. For more information on these calls, see *IMS Version 9: Application Programming: Design Guide*.
CCTL abended, looping, or in wait state	Use one of your own programs.
I/O error on SLDS or RLDS	Use an SLDS or RLDS that has previously been damaged. Reset, rewind, and reload tape drive.
MVS error (loop or abend)	Unplug or switch off MVS system residence drive. Cancel IMS control region.
Hardware error, no loss of real storage	Unplug or switch off MVS system residence drive.
Power failure, or hardware error with loss of real storage	Press System Reset, and re-IPL.

Qualities of Good Tests

The specific nature of your IMS use and installation setup will determine the details of your testing program. However, all good tests have these qualities in common:

- They are realistic.
- They are specific—addressing each one of the many possible operating situations.
- They involve the same people who operate IMS in actual, production-mode situations.
- Their results are verifiable.

Improvement through Feedback

Although formal testing of your procedures ends when production begins, each actual instance of operation and recovery can be considered a test—a chance to improve your procedures. If problems develop or procedures are found to be vague or inaccurate, you should use this experience to update and improve your procedures.

You might also want to consider conducting periodic reviews of your procedures by your operations staff and system administration staff. Such reviews are particularly important as you add maintenance to your IMS system or upgrade to a new release of IMS.

Retesting When the System Changes

Any time you significantly change your system, you should do at least some retesting. Changes could be in the form of either a new release or major upgrade of IMS itself, or a redesign or expansion of your production setup.

The retesting should verify that your old procedures still function properly in the new environment.

Chapter 18. Developing End-User Procedures

In addition to developing procedures for operating IMS, you also need to develop procedures for end users, sometimes called remote terminal operators. End users are the people who use the terminals connected to IMS. It is important that your end-user procedures be good ones, because the success of your online system depends largely on its acceptance by your end users.

End-user procedures should focus on the needs of end users, not on IMS operation. The procedures should be prepared as an integral part of the application-development process. The layout and organization of the procedures should support the business function. Procedures differ for different types of applications. Procedures for a bank teller, for example, should cover more than just the entry of an IMS transaction. Procedures for a data-entry clerk, however, may consist largely of instructions for entering transactions.

End-user procedures are generally packaged in a guide. You can develop the guide yourself, or it might be appropriate for individual application-development teams to develop the guide. In the latter case, you should consider preparing “boiler plate” material to distribute to the application-development teams for basic procedures that are always included in end-user guides. Do not forget to update the boiler plate information whenever you make major modifications to the online IMS system: for example, if you convert from using BTAM to using VTAM, you probably need to change terminal logon procedures.

In general, your procedures need to familiarize end users with how to:

- Operate their terminal
- Establish a connection to IMS
- Communicate with IMS
- Disconnect from IMS
- Respond to error conditions

The rest of this chapter describes these procedures and where you can find more information for them.

Procedures for End-User Terminal Operators

The application-development cycle should include the development of end-user procedures, along with education about the transactions and terminal hardware. The writers of the procedures need to consider the following while developing the end-user guide:

- Consistency between remote and central control
- Which services should be provided by the online system rather than the terminal or workstation
- Coordination of support for end-users
- Definition of problem-reporting procedures

Reviewing End-User Operator Tasks

An end user performs one or more tasks as part of his or her job. These tasks can involve many of your IT tools and systems, of which IMS is one. You should review those tasks to be sure your procedural instructions do both of the following:

- Help the user perform application-related tasks

- Safeguard the primary task

You should review existing instructions that have implications for administration services. Table 62 shows a partial checklist illustrating the characteristics of this review. You should expect other topics depending on the scope of the user procedure.

Your review contributes to the usability of the procedures. You should plan to correct them based on testing and user feedback during actual production. “Establishing Operating Instructions” addresses the overall structure and content of remote terminal operator procedures.

Table 62. Checklist for Administrative Services

Review Item	Related Administration Task
Are availability statements clear?	Check scheduling algorithm and operations procedures.
Are response claims reasonable and not misleading?	Reconcile with performance criteria. Ensure unusual delays are reported.
Are terminal connection procedures described?	Check whether the MTO or other operators help in connection.
Are security requirements explained?	Check security definition and maintenance procedure.
Is interpretative material included for output?	Check accuracy of IMS message and command output information.
Are instructions given for an alternative connection?	Check that the MTO has alternative configuration instructions.
Are instructions given for response to system and broadcast messages?	Check with developer on potential messages.
Are instructions given for service problems for the terminal, transaction input, and online system failure?	Coordinate failure reporting and information flow. Have MTO issue standard information messages.

Establishing Operating Instructions

The success and acceptance of an online system often depend on the quality of the end-user documentation. The operating procedures for terminal operators form an important part of this documentation, contributing to ongoing education and ease of use for the applications.

Develop and refine the procedures in several stages:

- Define initial structure and content
- Analyze existing procedures
- Test the new procedures
- Gather education feedback
- Gather production feedback
- Make any necessary modifications to the procedures

It is generally better to extract information from existing sources and tailor it to an application rather than point to total descriptions or general operation manuals. For example, describe the 3270 operational characteristics rather than assume the operator is familiar with the content of *3270 IDS Operator’s Guide 3275*.

Also, show representative screen formats rather than showing MFS output. You can obtain working copies of the screen layouts by using the copy feature on a remote 3277 terminal.

Do not ignore the last stage, the actual production cycle. In many cases, you can obtain useful feedback by selecting a group of end users to be pilot operators. Their mission is to validate and improve the end-user service. After making any necessary changes, publish the final procedures.

Do not neglect to update the procedures when you make modifications to the online IMS system. Sometimes the operational emphasis changes when you install subsequent application programs or IMS features. Your same pilot group could renew their activities when hardware changes or other significant application changes occur.

Some guidelines for the content of end-user operator documents follow. The goal is to have the document tuned to the experience level and information requirements of the end users:

- Make the main section procedural for the applications.

  Ensure the instructions are sequenced in the order in which they are performed. Place transactions that are more frequently used, or more important, ahead of others. If there are numerous transactions, an alphabetic order might be better, if you include an introduction (or graphic) that indicates which are critical or used frequently.
- Illustrate your procedures with simple examples.

  Show both entry and response formats. Where IMS commands are used, show only relevant keywords and explain the replies.
- Include instructions for error handling.

  Address not only input errors but also responses to warning messages from the application program. Where IMS connection or system problems might occur, provide alternative instructions. If this material is lengthy, you could place it in an appendix. Arrange the topics in the appendix by symptom. Provide an explanation of error-reporting forms.
- Optionally, include a brief introduction to IMS and describe the MTO.

  The emphasis should not be on how IMS works, but on how it supports data access and processing at your site. Distinguish, as necessary, between what IMS provides and what the operating system and hardware provide.
- Identify the main hardware operational characteristics.

  Include advice that highlights efficient ways of using the terminal and warnings of any user actions that are prone to error. If appropriate, merge copies of relevant operating instructions into this section or appended to the document.
- Include only relevant IMS command operation details.

  Make sure that commands and options described are allowed for end-user use. Discourage the use of commands that reserve IMS resources, such as `/HOLD` for conversations or `/IDLE` which enqueues output messages.
- Include a list of contacts.

  Show names in preferred order and describe the conditions under which the user should make contact.

Potential Use of IMS Commands

Although the primary purpose of end-user procedures is to describe transaction input, you might also allow the user to enter a subset of IMS commands so he or she can make more informed use of IMS resources. For example, if the end user can choose the order in which to submit transactions, he or she can use a /DISPLAY command to gauge the current queue levels.

Recommendation: If you do allow end users to enter IMS commands, consider disallowing the use of the ALL keyword to minimize the effect of these commands. See the KEYWD macro in *IMS Version 9: Customization Guide* for information on disallowing use of the ALL keyword.

Table 63 presents a list of IMS commands and evaluates their potential use by a remote terminal operator. The /DISPLAY TRAN and QUERY TRAN commands are especially appropriate. However, because of the many keyword choices for these commands, you should specify individual keywords. See Table 66 on page 381 for appropriate keywords. For convenient entry, include command abbreviations. See *IMS Version 9: Command Reference* for the full list of command and keyword synonyms.

The commands in the table marked with * are available to the remote terminal operator using default command security. To allow the remote terminal operator to use other commands, you need to declare the full list of commands for that terminal in the IMS Security Maintenance utility input or through RACF (or an equivalent product). You can also use the DFSCCMD0 exit routine for command security. For more information on this routine, see *IMS Version 9: Customization Guide*.

Table 63. Command Suitability for Use by a Remote Terminal Operator

Command Group	Command	Default Security	End-User Use	Comments
End-User Specific	/RCLSDST	*	YES	Acts as remote VARY offline (for VTAM terminals).
	/RCOMPT	*	YES	Decentralizes control of VTAM component.
	/BROADCAST	*	YES	Provides convenient user interaction; use of message switching is preferred.
	/SET	*	YES	Provides convenient entry for batched transactions.
	/RESET	*	YES	Used with /SET.
	/RDISPLAY	*	YES	Identifies MTO location; used with /BROADCAST.
	/CANCEL	*	YES	Cancels both single- and multisegment messages.
	/EXCLUSIVE	*	YES	Safeguards discrete application usage.

Table 63. Command Suitability for Use by a Remote Terminal Operator (continued)

Command Group	Command	Default Security	End-User Use	Comments
Connection Commands	/OPNDST		NO	Allows access to VTAM network.
	/CLSDST		NO	Restricts access to VTAM network.
	/QUIESCE		NO	Restricts access to VTAM network.
	/SIGN	*	YES	Allows access to IMS resources; required for signon verification.
	/RSTART		YES	Starts IMS resources; allowed if /STOP allowed.
	/MSASSIGN		NO	Alters assignments for IMS resources; required for MFS transactions.
	/FORMAT	*	YES	Formats a screen; required for conversation terminations.
	/EXIT	*	YES	Terminates conversations; required for non-VTAM switched terminals.
	/IAM	*	YES	Suspends conversations; gives flexible response.
	/HOLD	*	YES	Suspends conversations.
	/RELEASE	*	YES	Resumes conversations.
Connection Commands for MTO	/START or UPDATE		NO	Starts IMS resources.
	/STOP or UPDATE		NO	Stops IMS resources.
	/CHANGE		NO	Allows changes to mode table and automatic session restart. Immediate termination best handled by the MTO.
	/IDLE		NO	Allows control of multiple systems.
Device Control	/COMPT		NO	Sets terminal components; also available through /RCOMPT.
	/MONITOR		NO	Stops programmable remote stations, such as System/3 or System/7; best handled by MTO.
Monitoring	/DISPLAY or QUERY		YES	Displays information about IMS resources; convenient for status of queues and terminals.
	/TRACE or UPDATE		NO	Allows IMS to trace resource activity; IMS Monitor and traces affect system performance.
	/LOG	*	YES	Records user text on IMS system log.
Security	/DELETE		NO	Deletes security for IMS resources; restrict to MTO.
Testing	/TEST	*	YES	Sets test mode; limit use for remote terminal operators.
	/LOOPTEST	*	NO	Tests I/O for terminal; used for BTAM nonswitched problems.
	/END	*	YES	Terminates test, looptest, or exclusive mode.
	/MSVERIFY		NO	Verifies IMS resources; used by administrator of multiple systems.

Table 63. Command Suitability for Use by a Remote Terminal Operator (continued)

Command Group	Command	Default Security	End-User Use	Comments
Resource Control	/ASSIGN or UPDATE		NO	Assigns IMS resources; prime control of resources for MTO.
	/PSTOP or UPDATE		NO	Restricts use of resources.
	/PURGE or UPDATE		NO	Restricts input to resources.
	/LOCK or UPDATE	*	NO	Restricts resource use; use only for supervisor remote terminal operators.
	/UNLOCK or UPDATE	*	NO	Unlocks IMS resources.
	/RMxxxxxx		NO	Allows control of DBRC resources; database availability should be controlled by MTO.
	/SMCOPY		NO	Controls secondary master terminal.
	/MODIFY		NO	Allows online changes to IMS system.
Recovery	/DBRECOVERY or UPDATE		NO	Controls database availability; restrict to MTO.
	/DBDUMP or UPDATE		NO	Controls database availability; restrict to MTO.
	/DEQUEUE		NO	Controls integrity of message queues; restrict to MTO.
	/CHECKPOINT		NO	Shuts down IMS; restrict to MTO; simple checkpoints can have a performance impact.
	/ERESTART		NO	Restarts IMS; restrict to MTO.
	/RECOVER		NO	Recovers full-function databases and Fast Path areas; restrict to a DBA or the MTO.
/SWITCH			NO	Switches data sets used in an XRF environment; restrict to MTO.

Problem Reporting for a Remote Terminal Operator

When you examine the sequence of instructions for a remote terminal operator, you need to ask a series of “what-if” questions. The main flow of the end-user guide should present the normal operations sequence, but the operator needs to be prepared for unexpected or error responses. The following are examples of these “what-if” questions:

- What if the terminal is not operational, behaves erratically, or loses contact with the system?
- What if the application program returns warning messages about the status of a database or of processing results?
- What if IMS system messages are received?
- What if master terminal or other operator actions interrupt processing?
- What if the MVS or IMS system goes down?

You need to assess the importance of such unusual events and include appropriate procedures for them. The end-user guide could describe recovery actions inline with normal operations or include the information in symptom-based appendixes.

If your installation has a problem-reporting center, having a user-liaison group enables you to concentrate recovery skills rather than educating all end users. Many installations report terminal problems to network control personnel, and they have appropriate details about IMS and the application programs it runs. It is often a good idea to standardize problem reporting. You can prepare the answers to a series of questions, or elicit a problem's symptoms using a structured script.

Occasionally, the master terminal operator can be a point of contact. You should ensure such contact is only be on a formal, prearranged basis, because the master terminal operator has a primary duty of system control and monitoring. An example of acceptable contact is to request the scheduling of a BMP.

A good controlling technique is to require an error incident report. You can make preformatted forms, with examples of the information required, available to the end user. The data on the forms should indicate what action was taken or is pending. Administration personnel can then review these events on a regular basis. In this way, you can detect common problems or differences in operational procedures.

Operating the Terminal

Information on the operation of individual terminals is contained in the manuals for the terminals or devices. Appendix A, “IMS Support of Devices,” on page 355 describes IMS support of individual terminals and identifies, for each terminal, the appropriate manual. You might want to give the appropriate manuals to end users to familiarize them with a particular terminal's operation. Or you could extract the appropriate operational information and put it in the end-user guide.

Connecting to IMS

Before an end user can use a terminal, the MTO must initialize the communication network using one or more of the following commands: `/START DC`, `/START NODE`, and (possibly) `/OPNDST` for VTAM terminals, or `/START LINE` for BTAM terminals.

Related Reading: Appendix A, “IMS Support of Devices,” on page 355 contains device-specific information on how to establish a connection to IMS (basically how to log on) for different types of terminals.

Terminal Startup for Nonswitched Lines

For startup to be successful, the remote terminal must be turned on. IMS notifies the remote terminal of a successful `/START` or `UPDATE` command with the `DFS059` terminal started message. Communication with IMS can only begin after the startup message is received.

Exceptions: A remote terminal can communicate with the following terminals before receiving message `DFS059`: the master terminal, physical terminals on the master terminal line, and the system console.

Terminal Startup for Switched Lines

When a switched line starts, it can accept calls from remote terminals. The data set attached to the remote terminal must be used to establish a connection with IMS. After the call is made and a connection is established, the terminal operator uses the `/IAM` command to identify the terminal to IMS.

IMS notifies the remote terminal of a successful /IAM command with the DFS058 command completed message. Communication with IMS can begin only after this message is received.

Restriction: Fast Path transactions cannot be entered from a switched terminal.

VTAM Terminals (Nodes)

SLU type P, FINANCE (3600), and ISC terminal systems do not receive IMS-generated connect messages. Instead, they communicate with a user-written program that resides in the terminal system's storage. Type-1 SLUs in unattended mode also do not receive connect messages (because there is no operator to receive the message).

VTAM terminals receive message DFS3649, DFS3650, or both. The DFS3649 message requests signon information, and the DFS3650 message indicates successful signon (or connection) and session status. Message DFS3649 also appears in the following situations: when failures occur during the signon process, when the operator signs off, or (for ETO only) automatic session timeout due to inactivity.

Related Reading: For more information about these messages, see *IMS Version 9: Messages and Codes, Volume 1*. See *IMS Version 9: Administration Guide: Transaction Manager* to determine which message is received during the initial terminal connection processes.

Releasing VTAM Terminals Hung in I/O

Use the /TRACE command to specify what action, if any, IMS should take when it determines that a VTAM response has been outstanding for a longer time than you specify. This facility applies only to VTAM devices that have VTCBs or MSC links.

IMS can handle outstanding responses in one of the following ways:

- No action.
- Issue a message that I/O has been outstanding for a longer time than you have specified; that is, the device has timed out.

  In this case, the operator must determine what action, if any, to take. For example, the MTO or an AOI program could issue an /IDLE command followed by an /ACTIVATE command and an /OPNDST command.
- Issue a message that I/O is outstanding followed by a VTAM VARY NET,INACT command to deactivate the terminal, and a VTAM VARY NET,ACT command to reactivate it. If the device is operable and defined to IMS as non-shared, and IMS is not shutting down, IMS issues a /OPNDST command to establish a session with the node.

Use the /DISPLAY command to display all the nodes that have I/O which has been outstanding for a longer time than was specified during installation. The /DISPLAY TRACE command displays the status of all nodes that have I/O outstanding for a longer time than you specified using the /TRACE SET TIMEOUT command.

Related Reading: See *IMS Version 9: Command Reference* for information about using these commands.

Fulfilling Security Requirements

Depending on the security options specified for the system, the end user might need to use the /SIGN command to provide a user ID for verification by IMS or

RACF and supply an IMS or RACF password. Users with ETO terminals must use the /SIGN command. You need to document the security procedures the end user must follow.

Communicating with IMS

An end user can communicate with IMS in the following ways:

- Entering transactions
- Sending messages to operators or other users
- Using IMS commands

Entering Transactions

When using a terminal (a non-programmable workstation), the operator must enter a transaction in a certain format in order for it to be recognized by IMS. When the operator uses the /FORMAT command, IMS formats the screen using MFS. This format is defined by the installation.

Transaction Code Format

Use the transaction code format shown below to send a message to an application program:

```
transaction_code     (password)    text
```

where:

transaction_code
Is a 1- to 8-byte alphanumeric code defined during IMS system definition. Special characters (embedded blank, slash, dash, equal sign, comma, and period) are not allowed. A blank or left parenthesis must follow the transaction code. You can use the COMM system definition macro to eliminate the need for a blank in a segment consisting of only the transaction code.

(*password***)**
Is a password that must be entered with the transaction code (when required by installation security). Enclose the password in parentheses and concatenate it to the transaction code. If a password is not required, IMS ignores any password you enter.

For terminals using MFS, password entry is controlled by installation-defined formats.

text
Is the single-segment or multisegment input message. Only the first segment of a multisegment message can contain the transaction code and password.

Use the /SET command to send a continuous series of messages to the same application program. The /SET command puts the terminal in preset mode and eliminates the need to enter the transaction code (and password, if required) for each message. Preset mode fixes the destination for all messages entered from a terminal.

Documenting Transaction Input

The text included in the transaction is application dependent, and should be clearly documented for each transaction type an end user might use.

Using Response Mode

Response mode describes a connection between IMS and a communication line, terminal, or user that occurs for certain types of terminals or users under conditions specified during system definition. When the response mode is in effect, IMS does not accept input from the line, terminal, or user until IMS has sent an output response to the previous input. Response mode can be defined for either full function or Fast Path.

You can specify communication lines, terminals, or users as:

- Always operating in response mode
- Never operating in response mode
- Operating in response mode only if the transaction code being processed specifies it

Any transaction can be defined as a response-mode transaction. When IMS receives an input transaction that requires response mode, IMS makes the terminal is unusable (locks the keyboard) until IMS processes the transaction and sends the response.

While in response mode, if the user enters a single transaction for which the application program generates no response, IMS automatically sends message DFS2082 as the response to the transaction. This message removes the terminal from response mode and unlocks the keyboard, if necessary.

While in response mode, if the user enters multiple transactions for which the application program generates no response, the keyboard remains locked and the terminal is unusable. The terminal remains in response mode even if the user attempts to unlock the keyboard (by, for example, pressing the RESET key on a 3270 terminal); any subsequent input is rejected. The first output from an application program whose input originated from this terminal satisfies the response.

User IDs defined with ETO user descriptors remain in response mode after abnormal session termination or autosignoff. In this situation, the end user should contact the MTO (using a means other than the terminal or user ID). If the terminal is non-operational, the MTO can enter an /RSTART command to restore terminal operation.

Attention: If you restart IMS while users are in response mode, what happens to transactions depends on whether they have reached a sync point:

- If the transaction is complete (it has reached a sync point), and the end user is waiting for a response, the user does not receive the normal response because IMS restart resets response mode. After the restart completes, the user might reenter the transaction, which could adversely affect data integrity.
- If the transaction is incomplete (it has not reached a sync point), IMS discards the transaction.

You should develop operating procedures to cover this situation for all end users who could be affected.

Using Conversation Mode

Conversational processing allows an end user to have a continuing dialogue with an application program. A user initiates a conversation by entering a transaction code that is defined as conversational. A conversation can follow an ETO user from one terminal to another. The maximum number of conversations that can be active in the system at one time is 65535.

The following commands affect conversations:

- `/DISPLAY` displays the status of conversational processing in the system
- `/EXIT` terminates an active or suspended conversation
- `/HOLD` suspends an active conversation
- `/RELEASE` restarts a held conversation
- `/RSTART LINE | NODE | USER` recovers an active or suspended conversation on a stopped line, node, or user
- `/START LINE | USER` restarts the line or user but terminates the conversation

Using the appropriate keywords of the `/DISPLAY` command, the MTO can determine:

- The identification of all active (BUSY) or suspended (HELD) conversations in the system
- The number of conversations active or suspended in the system or related to a specific line, line and physical terminal, or user
- The number of inactive available conversations (the maximum number of conversations defined for the system minus the number of currently active or suspended conversations)

When you use the `/DISPLAY` command to display the status of a line or physical terminal, the displayed status includes, if applicable, the status of conversations related to that line.

Sending Messages to Other Operators

An end user can send messages to other operators by using the IMS message switch facility or by using the `/BROADCAST` command. If your end users need to send messages, document clearly how they are to do this.

When using message switching, the input format is as follows:

```
logical_terminal_name    text
```

where:

logical_terminal_name
Is the name of the logical terminal to which the message is to be sent. The logical terminal names allowed are specified during IMS system definition.

A blank must follow the logical terminal name. You can use the system definition COMM macro to eliminate the need for a blank in a segment consisting of only the logical terminal name.

text
Is the single-segment or multisegment input message

For a multisegment message, only the first segment should contain the logical terminal name. If the logical terminal name is specified in subsequent segments, it is treated as text.

You can use the `/SET` command to send a continuous series of messages to the same logical terminal. The `/SET` command puts the terminal in preset mode and eliminates the need to enter the logical terminal name for each message. Preset mode fixes the destination for all messages entered from a terminal.

Using IMS Commands to Communicate with IMS

Although the primary purpose of an end-user guide is to describe transaction input, you might allow end users to use certain IMS commands. If you do, you should document their use in the guide. Table 63 on page 342 presents a list of IMS commands and evaluates their potential use by an end user.

Command Format

IMS Version 9: Command Reference describes the IMS command language in detail.

Controlling Special Operating Modes with Commands

IMS provides commands to enter and exit special operating modes. These modes alter the relationship between the terminal and the system.

A special operating mode remains active until one of the following events occur:

- The user issues a command to exit from the special operating mode
- The MTO stops and restarts the line, physical terminal, or user by issuing the /START or UPDATE command

To restart a terminal or user without affecting its special operating mode, use the /RSTART command.

The /IAM command removes a terminal from any special operating mode.

Preset Mode: Preset mode fixes the destination for all messages entered from a terminal. Use the /SET command to enter preset mode. To leave preset mode, use the /RESET command.

Test Mode: Test mode (also called “echo” mode) performs input and output operations between a terminal and IMS. The /TEST command places a terminal in test mode. Messages entered from a terminal in test mode are transmitted back to that terminal. The /END command terminates test mode. In an ETO environment, the /TEST command applies only to ETO-defined users, not to ETO-defined terminals.

MFSTEST Mode: MFSTEST mode tests MFS control blocks without interrupting normal production activity. The /TEST MFS command places MFS-supported terminals in MFSTEST mode. The /END command terminates MFSTEST mode. In an ETO environment, the /TEST command applies only to ETO-defined users, not to ETO-defined terminals.

Looptest Mode: Looptest mode tests output operations between a terminal and IMS. Use the /LOOPTEST command to enter this mode. This command creates an output write loop that repeatedly transmits a single-segment message to the terminal being tested. The /END command terminates looptest mode.

Exclusive Mode: Exclusive mode prevents output messages from being transmitted to the terminal.

Exception: Output messages that are responses to messages entered at the terminal during exclusive mode **are** transmitted to the terminal.

After the terminal is removed from exclusive mode, the output messages are held for transmission. Use the /EXCLUSIVE command to enter exclusive mode, and the /END to leave exclusive mode. In an ETO environment, the /EXCLUSIVE command applies only to ETO-defined users, not to ETO-defined terminals.

Disconnecting from IMS

If an ETO-defined terminal or user ID is idle (no transactions are performed) for a given amount of time, IMS automatically disconnects the terminal or user ID. This amount of time is determined by the MTO. The end user can sign back on by issuing the `/SIGN` command.

Related Reading: For more information about autosignoff and autologoff, see *IMS Version 9: Administration Guide: Transaction Manager*. For device-specific information on how to terminate a connection to IMS (basically how to log off), seeAppendix A, "IMS Support of Devices," on page 355.

Responding to Errors

In addition to procedures, end users need some administrative support for responding to errors. This support can be provided in a variety of ways, one of which is to use a user-liaison group. A user-liaison group lets you concentrate recovery skills, rather than educating all end users. It is a good idea to standardize end-user problem reporting. You can have the end user answer a series of questions before calling the user-liaison group, or you can have the user-liaison group elicit problem symptoms from the end user using a structured script. Among other things, each report should include which system and terminal experienced the error.

No matter how you decide to provide administrative support, the end-user guide should include a "contacts list". This list identifies the people to be called for application, system, or terminal errors. Names, functions, and telephone numbers should be included in this list.

Related Reading: See "Problem Reporting for a Remote Terminal Operator" on page 344 for more information on problem reporting. See Appendix A, "IMS Support of Devices," on page 355 for information on specific terminal-related error conditions and how to respond to them.

Testing and Maintaining End-User Procedures

As with all procedures, end-user procedures should be carefully tested, and maintained whenever the system changes. The general remarks on these topics in Chapter 17, "Developing Operating Procedures," on page 307 apply equally here.

Part 4. Appendixes

Appendix A. IMS Support of Devices

This appendix describes the devices that IMS supports. This information should help you develop those parts of your end-user procedures that are device related. This appendix assumes you are familiar with the general features and operation of each device. The focus is on device operation when you use IMS.

3270 Information Display System

During IMS system definition, you can specify that a 3270 system attached to IMS should operate in forced-response mode, nonresponse mode, or transaction-dependent response mode. For an ETO terminal, specify these modes in the ETO user descriptor.

When 3270 terminals are **not** attached to IMS through VTAM, the MTO can initiate their use by entering the appropriate `/START LINE` command for the lines to which the 3270 components are attached.

When 3270 terminals **are** attached to IMS through VTAM, they are initiated when one of the following occurs:

- The MVS VTAM network operator enters a `VARY NET` command (for example, `VARY NET,ID=`*a*`,LOGON=`*b*).
- The IMS MTO enters the `/OPNDST` command.
- If the 3270 components are defined in the VTAM definition as belonging to IMS, VTAM makes them automatically available.
- The 3270 terminal operator enters a request to VTAM to be connected to IMS.

Each operational terminal receives a `DFS059I` message indicating that the terminal has been started. The MTO can use a `/DISPLAY` command to determine which lines are attached.

Related Reading: For information on the general features and operation of the 3270, see *3270 IDS Operator's Guide 3275*.

Connecting to IMS (3275 with Dial Feature-Switched Line)

To connect a 3275 with the dial feature-switched line to IMS:

1. Make the 3275 ready (if the terminal is off, power it on; if it is already on, power it off and back on).
2. Dial the IMS telephone number on the data phone.
3. Switch to data mode when you receive the correct answer tone.
4. Press the 3275 Clear key.
5. Enter the `/IAM` command when you receive the `DFS073I` message.

When IMS accepts the `/IAM` command signon, the 3275 is physically and logically attached to IMS. From this point on, operate the switched 3275 in the same way as a nonswitched 3275.

Disconnecting from IMS (3275 With Dial Feature-Switched Line)

To physically disconnect the 3275 from IMS, enter the `/IAM DONE` command, or raise and release the 3275 Disconnect switch, then place the handset on the hook when the System Ready light goes off.

Communicating with IMS

After you receive either message DFS059I or DFS3650I telling you that the terminal or session is started, you can enter a transaction code, a command, or an IMS message switch.

If you enter data with a keyboard, a selector light pen, a magnetic card reader, or program function key special features, use the /FORMAT command to display a format on the screen before you begin working with IMS. When you enter a transaction code, IMS returns and displays the output data from the application program that processes that transaction. This output might include a prompt to solicit additional input. You can fill in only the blank fields on the screen. After entering data, press Enter to send it to IMS for processing by the application program. Continue in this manner until the interaction with IMS or the application program is complete.

Operating the Terminal

This section contains information on operating 3270 components with IMS.

Display Size

The 3270 display station can display from 480 to 3564 characters of information, depending on the model.

The 3275 or 3277 terminal can display:

- 480 characters (Model 1)
- 1920 characters (Model 2)

The 3276 or 3278 terminal can display:

- 480 or 960 characters (Model 1)
- 1920 characters (Model 2)
- 1920 or 2560 characters (Model 3)
- 1920 or 3440 characters (Model 4)
- 3564 characters (Model 5)

The 3279 terminal can display:

- 1920 characters in 4 colors (Model 2A)
- 2560 characters in 4 colors (Model 3A)
- 1920 characters in 7 colors (Model 2B)
- 2560 characters in 7 colors (Model 3B)

These characters can be alphanumeric characters, special characters, certain device-control characters (visible on the screen), and attribute bytes (not visible on the screen). In addition, you can prevent display of information in certain fields on the screen.

Page-Advance Function

A transaction’s output can be more than one full screen of data. You request the additional output by pressing the PA1 (program access 1) key on the terminal keyboard (see “Operator Logical Paging” on page 357).

If you continue to press the PA1 key, the terminal displays successive pages until you reach the end of the current message. If you press PA1 when there are no more messages waiting to be sent to the display station, the display remains unchanged or IMS sends an appropriate error message.

Message-Advance Function

If you press the PA2 key at any time during a message, IMS deletes the current output message and displays the next message. If no messages are waiting when you press the PA2 key, IMS does not modify the screen.

Password Entered from 3270 through MFS

Entering a password through MFS can be different from password entry on a cleared screen or on other IMS terminals. The password can be created from:

- Data read from a 3270 ID card reader
- Data that you key into one or more fields or a portions of fields in the device format
- Data that you key into a field that is not displayed on the screen and is designated as the password field only in the device format
- Literals
- Combinations of the above

Operator Logical Paging

If you do not specify operator logical paging, the terminal operator can advance pages only one at a time, using the page advance function. If you do specify operator logical paging in the application program's message output description, the terminal operator can page forward or backward to any page in a message.

With operator logical paging, the terminal operator can enter any of the functions shown in Table 64 to display a particular logical page.

Table 64. Functions for Operator Logical Paging

Enter	To display
=	The next logical page
=*n* =*nn* =*nnn*	The *n*th logical page of the message
=+*n* =+*nn*	The *n*th logical page after the current page
=−*n* =−*nn*	The *n*th logical page before the current page
=*L*	The last logical page of the message

Enter these functions in the screen area reserved for command and page request information. If no area is reserved, either clear the screen (using the Clear key) and then enter the paging request, or use the next logical page (NEXTLP) function if it has been defined as a PF key in the format definition.

If you request a page outside the range of the pages in the message, IMS issues an error message. After the error message displays, press the PA2 key to continue the message that was being displayed when the error occurred. If a logical page in the message is larger than one physical page, use the PA1 key to step from one physical page to the next within a logical page.

Modified Data Tags

IMS resets modified data tags each time it unlocks the 3270 keyboard.

Exceptions: IMS does not reset modified data tags when:

- IMS sends a message using the SYSMSG field.

- IMS sends output to its master terminal using the MFS master-terminal format. IMS resets modified data tags only after you enter input.

Screen Protection

Under certain conditions, IMS avoids sending output to a 3270 terminal that could possibly destroy the operator's current work. IMS protects the screen so you can safely enter data or control requests when you:

- Set the screen mode to protect and:
  - IMS successfully sends output to the screen.
  - You press the Clear key and IMS responds by unlocking the keyboard.
  - You or the application program perform a successful Copy function.
- Requested the NEXTMSGP function (the PA3 key or a PF key so defined) when no output message was available.

The screen is not protected after you:

- Press one of the following keys:
  - Enter
  - PA1 or PA2
  - Any PF key (except a PF key defined as NEXTMSGP)
- Operate either of the following devices:
  - The selector light pen
  - The ID card reader

Attention: If the screen is protected, an IMS shutdown checkpoint cannot complete.

During IMS system definition (using the TERMINAL and TYPE macros or ETO logon descriptors) you can specify that a 3270 screen be left in unprotected mode when IMS sends an output message to it. In unprotected mode, IMS can send output to the device at any time without requiring input from the terminal (an input message or a PA1 or PA2 action). IMS provides this option on a terminal-by-terminal or message-by-message basis. The terminal option of unprotected mode applies to all messages for that terminal. If you do not select unprotected mode, MFS-formatted messages leave the screen protected or unprotected on a message-by-message basis, and messages that bypass MFS leave the screen protected.

By allowing the terminal to be unprotected, IMS can continually update the 3270 display terminal with current information without requiring the terminal operator to initiate any action to receive output messages. Use this option only with terminals that require limited input and require the most current IMS output message.

To avoid losing data, press the PA3 key (if the PA3 key is not used for copy) before entering data. When you press the PA3 key, you put the terminal in protected mode, and you can enter data. After you enter the data, IMS returns the terminal to unprotected mode.

Lock and Unlock Option

During IMS system definition (using the TERMINAL and TYPE macros or ETO logon descriptors) you can specify whether the application program or IMS should be responsible for unlocking the terminal keyboard and resetting the modified data tags. If you select `LOCK`, the application program unlocks the keyboard after it receives input. If you select UNLOCK, IMS controls the unlocking of the keyboard. This option only applies to messages that bypass MFS on output with a MOD name of DFS.EDTN.

If the application program terminates without unlocking the keyboard, you can unlock the keyboard by pressing the Reset key. If the application has not terminated but is slow in responding, do not press the Reset key and start entering data, because the data might be lost when the application program sends another output message to the terminal.

3270 MFS Bypass Option

If you select the 3270 MFS bypass option (by using a MOD name of DFS.EDTN), the application program controls the functions performed by the Clear key, selector pen attention, PA1, PA2, or PA3 keys.

Programmed Symbols

Programmed symbols (PSs), an optional feature of the 3270 terminal, allow storage and access of up to six sets of 190 characters, each of which can be user-defined and loaded by the program.

You can load PS buffers with either a VTAM application or a user-defined procedure. After the PS buffers are loaded, you can use them on a field-by-field basis on MFS-formatted screens.

If a hardware error occurs while you are loading a PS buffer, IMS:

1. Returns the message used to load the PSs to the IMS message queue.
2. Takes the terminal out of service (except for SLU-2 devices).
3. Logs the error to the IMS log.
4. Sends a message to the IMS master terminal.

After you correct the hardware error and the terminal is in service, IMS re-sends the message used to load the PSs.

If the PS load function failed because of an error in the message used to load them:

1. Use the /DEQUEUE command to dequeue the message. You might have to enter the /DEQUEUE command from the master terminal.
2. Correct the error.
3. Reenter the transaction to send the message to reload the PSs.

For SLU-2 devices, if you have not loaded the PS buffers and IMS sends a message requiring the use of a PS buffer, the 3270 terminal rejects the message and IMS:

1. Returns the invalid message to the IMS queue.
2. Logs the error to the IMS log.
3. Sends a DFS2078I message to the node, telling you that the output was rejected. However, if the message being rejected is in storage, IMS sends the DFS2078I message to the IMS master terminal and closes the node.

IMS leaves the terminal in protected mode after it receives the DFS2078I message.

Screen Formats

The 3270 can operate with either an unformatted or a formatted screen.

Unformatted Mode: The screen is in unformatted mode:

- When you turn on the display station
- Whenever you press the Clear key on the keyboard

While the screen is in unformatted mode, you can enter the following types of messages:

- IMS commands
- Test requests (for BTAM terminal tests)
- Paging requests (when a message using logical paging is in progress)
- Transaction or message switch data that does not require MFS

Formatted Mode: IMS MFS defines the 3270 screen to have various fields with predetermined uses. The layout of the fields on the screen and the data displayed depend on the application program. A format consists of all the fields on a screen, and each message sent to a 3270 terminal must indicate a specific format.

An application program might use several types of formats. The format on the screen can change between messages, between input and output sequences of a message, or between transactions.

Different formats define input areas at different locations on the screen. When the 3270 terminal receives a message, the format usually places the 3270 cursor at the beginning of the first input area. The input can consist of commands, transactions, or message switches.

IMS-Supplied Input Formats: Some IMS-supplied formats allow one or two segments of input from the 3270 screen (one segment equals one line on the screen). When you want more than two segments of input, you can use the `/FORMAT` command to specify the `DFSM04` format. This format is an IMS-supplied format that allows up to four segments of input data. Each segment in the format consists of only the data entered without new line characters or other cursor-control characters.

Installation-Defined Formats: The installation can develop a variety of formats. Each format definition determines the information that you can enter and the way that you can use the format.

Installation-defined formats allow any of the following (either single-segment or multisegment):

- Transactions
- Page requests
- Commands

If the current installation-defined format does not allow the necessary input, try using the page-advance function (PA1), the message-advance function (PA2), or the Clear key to produce a screen format that allows entry of the desired data or command. Remember that using PA2 discards the remaining portion of the current output message.

Related Reading: For more information about either IMS-supplied or installation-defined formats, see *IMS Version 9: Administration Guide: Transaction Manager*.

Multiple Physical Page Input

An input message can consist of more than one physical page. For example, a 3275 or 3277 Model 1 display (480 characters) requires four physical pages for an input message designed to be equal to the 3275 or 3277 Model 2 screen size (1920 characters).

Use the following operational sequence as a guide when preparing procedures for 3270 operators who will use MFS and enter multiple physical pages of data to create one message. Figure 31 illustrates the procedure.

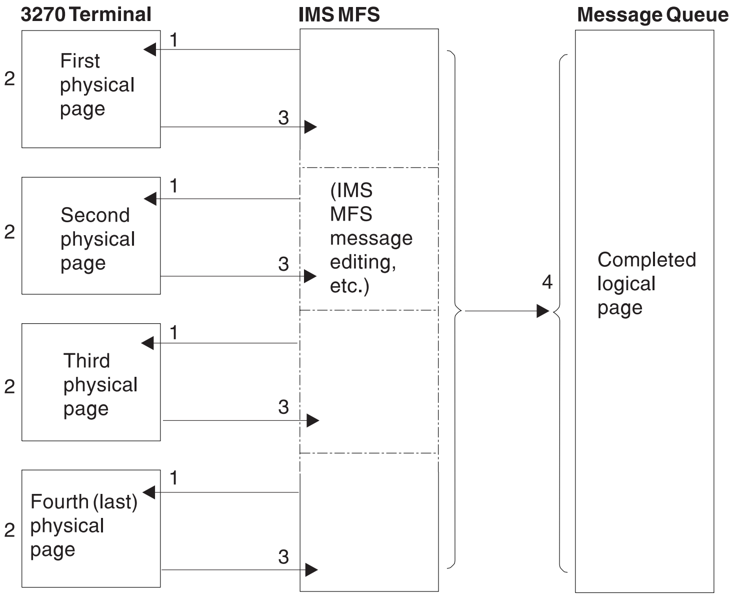


Figure 31. Data Entry Sequence for Multiple Physical Pages

Notes to Figure 31:

1. IMS sends the data-entry format for the physical pages to the 3270 screen.
2. The 3270 terminal is ready to accept your input whenever it displays the format of a physical page. You can input data (to start or continue creating a message) or use control functions.

   While the 3270 terminal displays the first physical page of an output message, your action determines what operation IMS performs:

   a. You can enter a message or choose not to enter a message.
      - When the 3270 terminal displays the first physical page and before you signal data entry, the following control functions are valid:
        – The PA1, PA2, and PA3 keys
        – The Clear key
        – The NEXTPP, NEXTMSG, NEXTMSGP, and PAGEREQ functions

        After you signal data entry, the rules for normal message creation apply.

   b. When you do not signal data entry on the first physical page, IMS assumes a display-only operation. When the 3270 terminal is displaying the first physical page, MFS can create an input message only if an output message is still available. An output message is not available if your previous request dequeued the current message and no other messages are enqueued. When IMS displays any page other than the first physical page on a display-only operation, only the following control functions are valid:

- The Clear key
- The PA1, PA2, and PA3 keys

Attempting any other function results in an error message.

c. You can create an input message and call for entry of that data:

Enter the required data into the physical page format displayed on the screen and then press Enter. Pressing a PF key that has not been assigned another specific function also causes the data to be entered.

You may want to advance to the next physical page without entering data from the currently displayed page. When the displayed physical page is not the first physical page, press the PA1 key.

d. You can signal the end of the current input message by entering the ENDMPPI function.

MFS edits the data keyed onto the screen and then performs end-of-message procedures to send the message to the message queue.

e. You can cancel the current input message by pressing one of the Clear, PA2, or PA3 keys, or by entering one of the NEXTLP, NEXTMSGP, or PAGEREQ functions.

IMS performs the respective function and discards all previous input for the current message.

3. You signal that the data on the screen is to be entered by pressing the Enter key.

   MFS accepts the data from the 3270 terminal and performs any message editing that is required (inserting fill characters, literals, and so forth).

4. MFS sends the completed message to the IMS message queue.

   You signal the end of the current message by entering the ENDMPPI function. MFS performs all required message editing and end-of-message procedures.

Additional rules that you must be aware of when you operate a 3270:

- You can enter the following from any physical page, but only once per message:
  - A PFK literal
  - An immediate pen-detect literal
  - An Op-ID card
- If you define DFLDs (device fields) with the same label on more than one physical page, MFS accepts data from only the first one defined as a part of the message.

3270 Master Terminal Support

A 3270 terminal can be the IMS master terminal. A 3270 master terminal consists of two 3270 devices: a 3270 display (3275, 3276, 3277, or 3278) and a 3270 printer (3284, 3286, 3287, 3288, or 3289). IMS defines the 3270 display as the primary master terminal and the 3270 printer as the secondary master terminal. IMS does not support the 3284 Model 3 as a master terminal.

When the IMS master terminal is a 3270 terminal, IMS sends all system messages to the primary device. IMS also sends selected system-generated messages critical to IMS operation to the secondary device. For more information about this topic, see the description of the COMM macro in *IMS Version 9: Installation Volume 2: System Definition and Tailoring*. IMS responds only to the primary device in the following situations:

- When you enter erroneous information, such as a nonexistent transaction code or an invalid parameter, from the master terminal

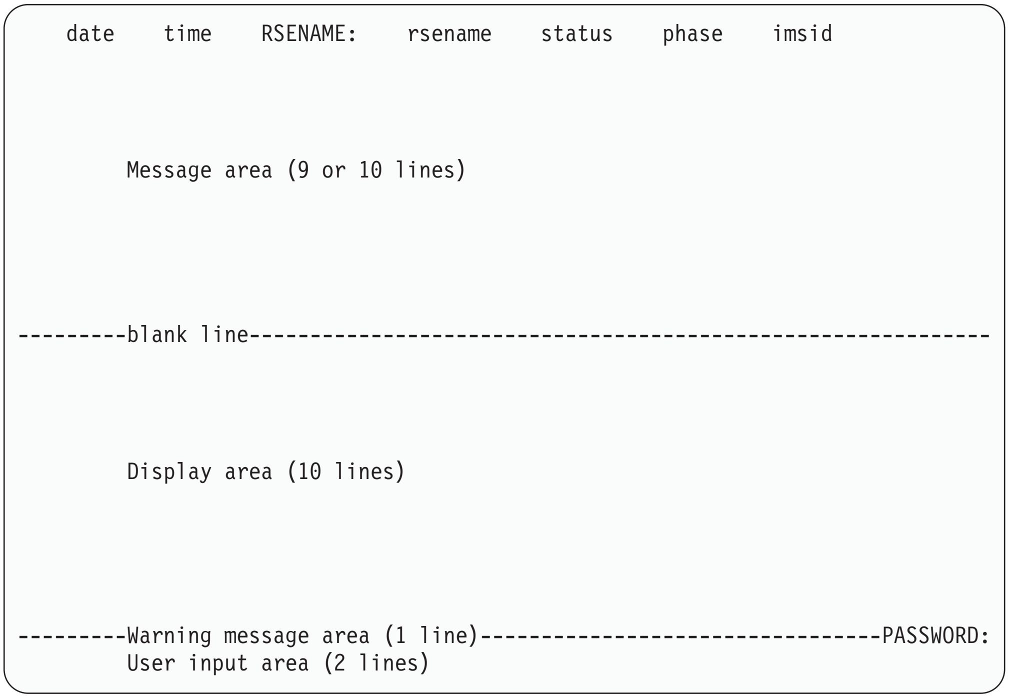
- When another operator switches a message specifically to the primary device
- When an application program inserts a message to the primary device
- When IMS issues a command-completed message or output from a /DISPLAY command
- When you reassign a 2740 master terminal to a 3270 display with a 3270 printer

If the secondary device is non-operational, IMS saves enqueued messages for transmission when it becomes operational.

The /ASSIGN command allows the 3270 master terminal to be assigned to a 2740 terminal and back again. The /ASSIGN command also allows independent assignment of the 3270 display and 3270 printer.

Operation Using the MFS Master-Terminal Formatting Option: When the IMS master terminal is a 3270 device, IMS displays system messages either with IMS-supplied formats (described in “Screen Formats” on page 360) or with the MFS master-terminal format.

The MFS master-terminal format is optional and is not available for all 3270 display devices. Choosing a 3270 display with 24 lines by 80 columns (a 1920 character display such as a 3275, 3276, 3277, or 3278 Model 2) as the IMS master terminal allows use of the MFS master-terminal formatting option. When you use this option, MFS divides the 3270 master-terminal screen into four areas: a message area, a display area, a warning area, and a user input area. Figure 32, which shows the relationship of the areas on the screen, is followed by a description of each area.



```
    date    time    RSENAME:    rsename    status    phase    imsid




         Message area (9 or 10 lines)





---------blank line------------------------------------------------------------





         Display area (10 lines)





---------Warning message area (1 line)---------------------------------PASSWORD:
         User input area (2 lines)
```

Figure 32. MFS Screen Format of 3270 (24-Line, 80-Column) Master Terminal

Status line: If the IMS subsystem is XRF capable, the top line of the screen displays the following fields in high intensity:

date	The current date
time	The current time

RSENAME: *rsename* — The name specified for the RSENAME parameter in the DFSHSBxx member of IMS.PROCLIB

status — The current status, which can be null, ACTIVE, or BACKUP

phase — The current XRF phase, which can be any of the following:

Null status	ACTIVE Status	BACKUP Status
INITIALIZATION	null AWAITING I/O PREVENTION	AWAITING SNAPQ SYNCHRONIZATION TRACKING IN PROGRESS TAKEOVER REQUESTED TAKEOVER IN PROGRESS

imsid — The IMS ID of the active subsystem

For more information about these items, see *IMS Version 9: Command Reference*.

Message Area: MFS designates the first ten lines of the screen as the message area if the subsystem is not XRF capable, or the first nine lines of the screen under the status line, if it is XRF capable. These lines receive such messages as:

- Responses to IMS commands (except /DISPLAY, QUERY, and /RDISPLAY)
- Messages sent from other terminals or applications
- IMS system messages

When the message area is full and more data is ready for output, MFS sends a message in the warning message area. When you signal readiness to receive output, IMS displays the new data on the top line in the area, overriding the previously displayed data, and inserts a blank line following the new line to separate it from the old messages.

Display Area: The 10 lines below the message area are the display area. MFS displays the output from the /DISPLAY, QUERY, and /RDISPLAY commands here, one page at a time. Each new page of displayed data completely replaces the previously displayed page.

Warning Message Area: Line 22 is the warning message area. The four possible warning messages issued by MFS are:

MASTER LINES WAITING
Displayed when you receive a multisegment message that is larger than the currently available message area, or when the message area is full and additional lines need to be displayed. To display the remaining lines, press the PA1 key.

MASTER MESSAGE WAITING
Displayed when the message area is full and you receive a message from another terminal or from an application program. To display the message and reset the warning, press the PA1 key.

DISPLAY LINES WAITING

Displayed after the display area is full because the output of a /DISPLAY, QUERY, or /RDISPLAY command is more than 10 lines. To continue displaying the output, press the PA1 key.

USER MESSAGE WAITING

Displayed when you receive a message for the master terminal that is not formatted in one of the IMS default formats. To display the message (in its format), press the PA1 key. To return the screen to the master format, enter any IMS command, or place the terminal in unprotected status and wait to receive a message in the master format.

The warning message area contains a literal, PASSWORD, followed by a field in which you can enter an IMS password. To enter a password, position the cursor immediately after the colon (:). You can press the back tab key to move the cursor from the user input area to the password field.

User-Input Area: The two bottom lines (lines 23 and 24) on the screen are the user-input area. You can enter up to two 79-character segments of input. When you are not entering data, the cursor is located at the beginning of this area (column 2 of line 23). MFS returns the cursor to this location when a response is received for the entered data. MFS leaves the entered data on the screen and you overlay it with your next input.

A 3270 master terminal using the MFS master-terminal formatting option is normally in unprotected status. This means that IMS can send output to the terminal at any time. You can put the terminal in protected status by pressing the PA3 key or the PFK11 key before entering input data.

The MFS-supplied master-terminal format defines literals for nine of the 3270 program function (PF) keys. The PF keys, when used with the 3270 master terminal formatting option, generate specific IMS commands. Pressing a PF key inserts a command into the first segment, in front of the data. The commands associated with the nine PF keys are:

PF Key	Command/Function
1	/DISPLAY or QUERY
2	/DISPLAY ACTIVE
3	/DISPLAY STATUS or QUERY STATUS
4	/START LINE
5	/STOP LINE
6	/DISPLAY POOL
7	/BROADCAST LTERM ALL
11	NEXTMSGP
12	Copy (3270 remote)

When pressing the PF7 key to enter /BROADCAST LTERM ALL, make sure the first line of input in the user-input area contains only the command. The second line of input should contain the data to be broadcast.

When you use the master-terminal format, MFS displays any message whose MOD name begins with `DFSMO` (except `DFSMO3`) in the message area. MFS displays any message whose MOD name is `DFSDSP01` in the display area. Messages with other MOD names cause the warning message `USER MESSAGE WAITING`.

Copy Function

You can use the IMS Copy function to obtain a printed copy of the currently displayed screen on a remote terminal. You can define the Copy function for one or more terminals on a remote 3270 line. For the complete description of devices that support the Copy function, see *3270 Information Display System, Configurator*.

You or an application program can use the Copy function through the use of the system control-area field. If you want to copy a screen from a display station, and the Copy function is allowed for your terminal, press the PF12 key.

The rules that govern the Copy function are:

In the SNA environment:

For SNA-defined nodes (in other words, SLU 2), IMS sends a copy request to the display node. The controller that selects the printer processes the copy command. The printer need not be defined to IMS.

In the non-SNA or bisync environment:

- If you request a copy from a 3275 display station, and a 3284 Model 3 printer is attached, the 3284-3 produces the hard copy.
- If you attempt a copy from a 3275 display station, and the 3284-3 printer is not ready, IMS issues an appropriate error message.
- If you request a copy from a 3277 display that is attached to a 3271 or 3274 control unit, or from a 3278 display attached to a 3274 control unit, or from a 3276 display (Models 1 through 4), IMS selects one of the following printers to receive the listing:
  – The printer that, during IMS system definition, you selected as eligible to receive copies from a specific 3270 display.
  – The first available printer of a group of 3270 printers that, during IMS system definition, you selected as eligible to receive copies from a specific 3270 display. If all printers are currently busy, the keyboard on the display station remains locked until a printer that can be used for the copy becomes free.
- If you attempt a copy from a 3270 display station attached to a 3271 or 3274 control unit or from a 3276 display station and no printers are currently available (all printers are stopped, pstopped, in exclusive status, or require intervention), IMS issues an appropriate error message.
- If you want all copy-function listings to be directed to a specific printer, you can use the `/STOP` command to prohibit the use of the other printers, or you can turn off the other printers.

  Restriction: The IMS copy function in the non-SNA/Bisync environment is not supported for ETO terminals.

To terminate a Copy function before it has completed, press the Reset key followed by either the Clear key or the Enter key. IMS terminates the copy function and unlocks the keyboard.

Function Keys

Certain keys on the display station keyboard are reserved for use by IMS. If you press any of these keys (or the selector light pen or operator ID reader) when the terminal is not screen protected, IMS might not be able to recognize your action.

Recommendation: Use these keys **only** when the screen is protected from receiving output.

These keys and their functions are:

PA1 — Page advance

The page advance key requests IMS to display the next screen of the current output message. The next screen can be the next physical page or the next logical page. The page-advance function is the only method of advancing from one physical page to the next physical page when a logical page consists of more than one physical page.

PA2 — Message advance

The message-advance key requests IMS to display the next message enqueued for the terminal. IMS deletes the current message, even if you have not displayed every page of it. If no more messages are waiting for the terminal, IMS leaves the current message on the screen.

PA3 — NEXTMSGP

The PA3 key usually performs the NEXTMSGP (message-advance protect) function, but could instead perform the Copy function.

PF12 — Copy

PF12 performs the Copy function for terminals that allow this function. The Copy function causes IMS to print the current display. See “Copy Function” on page 366. IMS does not restore keyboard operation until it has either transferred the message to the 3270 printer (3276, 3277, or 3278) or completed the printing (3275). After a copy has completed on a 3275, reposition the cursor to an appropriate location before attempting any input.

Clear Key

To set the display screen to unformatted mode, press the Clear key once. For information about the types of information that can be entered while in unformatted mode, see “Screen Formats” on page 360. While the screen is in unformatted mode, it is protected from receiving other output until you perform an input operation.

If IMS currently displays an output message, pressing the Clear key does not dequeue the output message.

If you press the Clear key when the terminal is not screen protected, IMS might not recognize the Clear request. However, if IMS does recognize the request, it unlocks the keyboard, displaying no data. If you press Clear and IMS responds by displaying data, the screen is invalid for input. You must press the Clear key again.

Enter Key

The Enter key sends to IMS either the fields currently modified by the operator or the last output message. The keyboard remains locked until IMS receives the message and restores keyboard operation.

3284 Model 3 Printer Operation

When attached to a 3275 terminal, IMS supports the 3284-3 printer as a component of the 3275 terminal. IMS considers the display portion of the terminal to be component 1 (COMPT 1) and the printer portion to be component 2 (COMPT 2).

As with any component-type terminal, IMS sends messages to the two components on a rotating basis. If IMS cannot send messages to the printer component, it sends messages continuously to the display component, as if no printer component existed. The operator must intervene to allow printer output. If IMS cannot send messages to the display component, it sends messages to the printer as if the display component did not exist. As long as IMS can send messages to the printer component, no operator intervention is required. Any situation (such as a stopped LTERM or non-operational printer) that prevents sending messages to the LTERM or LTERMs assigned to a particular component causes IMS to stop sending messages to that component.

To suspend messages to either the printer or the display component, use the /RCOMPT command with the NOTRDY keyword. This command sets the component that you specify as non-operational (not ready) if the 3270 is attached using VTAM. For BTAM-attached 3270s, use the /LOCK command to stop all LTERMs associated with the particular component.

Responding to Errors

This section describes how to respond to 3270-terminal errors.

Remote 3270 Errors (BTAM or VTAM)

A remote 3270 can inform IMS about certain unusual conditions that it detects during its operation. When IMS receives a message of this type, called a sense-status message, from a 3271, 3274, 3275, or 3276, IMS determines if it indicates an error or some normal condition. If the message indicates an error, IMS formats the message for printing and sends it to the MVS system console.

The sense-status messages, in general, indicate the following 3270 conditions:

- Device end (for example, operation complete on printer)
- Intervention required (for example, out of paper or 3270 display power off)
- Busy (operator action with unprotected screen)
- Hardware-detected errors (for example, data check or control check)

An intervention-required condition can be caused when:

- A display station is turned off (not on 3275)
- A printer is turned off or has its covers open
- A printer is without paper
- A security keylock is in the off position

When IMS detects an intervention-required condition, it suspends output to the unit. When you make the unit ready (by turning power on or by putting paper in the printer, for example), IMS resumes output. If a message is in progress when IMS detects the intervention-required condition, IMS resends that message from the beginning. For a 3270 printer, IMS can only detect an intervention-required condition that is caused by a power-off condition when IMS attempts to select that device for printing.

To determine which printers and displays are in an intervention-required state on a remote 3270 line, enter a /DISPLAY LINE PTERM or /DISPLAY NODE command. Those units in an intervention-required state display as COMPINOP (component inoperative).

IMS handles the printer on the 3275 (3284-3) differently when it detects the intervention-required condition. IMS marks the printer as having a component inoperable. However, the 3275 display continues to receive output messages and retains all input capabilities. Operations on the 3284-3 printer resume when you make the printer ready.

IMS retries 3271, 3274, 3275, and 3276 hardware-detected errors several times before it considers the error to be permanent. If you do not correct the error, IMS sets the physical terminal to pstopped and inoperative status and considers it unformatted.

IMS tries to pinpoint the cause of any hardware failures. It tries to eliminate the smallest component possible (for example, a terminal rather than a control unit, or a control unit rather than a line) when a problem forces elimination of resources. For a BTAM-attached 3270 terminal, the failing resource could be one display station or printer, a 3271 or 3274 control unit, or the entire communication line. In each instance, IMS marks all physical terminals belonging to the failing unit as pstopped and inoperative. For a VTAM-attached 3270 terminal, IMS issues a /CLSDST command to disconnect the terminal. After you make appropriate repairs, you can restore IMS operation by issuing a /START LINE PTERM or /RSTART LINE PTERM command for those terminals that are part of the failing unit.

IMS tries to ensure that a terminal correctly receives all data sent to it. However, certain printer operations, including the Copy function, cause mechanical actions on the printer after the data has been transferred. For some errors occurring under these circumstances, no error recovery is possible, and none is attempted.

Remote 3270 Error Recovery (BTAM)

For remote 3270 terminals attached to BTAM, IMS relies on the access method to retry errors occurring on the communication line. If the recovery attempt fails, IMS:

- Writes a diagnostic error record (X'6703') to the system log
- Sends a DFS051I message to the IMS master terminal
- Attempts to restart the physical terminals on the line

Because of the nature of transient communication-line errors, the attempt to restart the line and physical terminals is more effective if IMS waits for the communication line to quiesce before issuing the restart. Before issuing the restart, IMS waits for the interval specified in Table 65.

Table 65. Time Intervals for IMS Line and Physical Terminal Restarts

Type of Error	First Attempt	Subsequent Attempts
Timeout Error	10 seconds	20 seconds
All Other Errors	5 seconds	15 seconds

If the error persists after three restart attempts, IMS marks the physical terminals stopped and inoperative and issues either a DFS251I or DFS253I message to the master terminal, indicating the error condition.

Local 3270 Errors and Recovery (BTAM Only)

IMS makes all local 3270 terminals with permanent I/O errors pstopped or inoperative. To restart the terminal, enter a /START LINE PTERM or a /RSTART LINE PTERM command.

On some occasions, you may want to use a local 3270 line with missing DD statements. IMS automatically marks the inoperative terminals and attempts no I/O operations on them.

33/35 Teletypewriter

This section describes IMS operations for automatic send/receive (ASR) teletypes on a switched line.

Connecting to IMS Using the 33/35 Teletypewriter

Establish a connection to IMS using the 33/35 teletypewriter as follows:

1. Press the Orig button to receive a dial tone at the originating terminal.
2. Dial the number. The computer should answer after no more than three rings and respond with `IMS READY`.
3. Enter the `/IAM` command, unless you defined an automatic signon option during IMS system definition.
4. A successful signon results in the `DFS058I` IAM COMMAND COMPLETE message. You can enter messages from the keyboard. To enter data from paper tape, first enter:

   ```
   /RDR  CR,LF,X-OFF
   ```

Response Mode

During IMS system definition, you can specify that a 33/35 teletypewriter attached to IMS should operate in forced-response mode, nonresponse mode, or transaction-dependent response mode.

Communicating with IMS Using the 33/35 Teletypewriter

To send single-segment messages to IMS, use the following format:

- Begin the message with a transaction code.
- Terminate the message with the CR, LF, and X-OFF characters.

Example:

```
transaction_code  message_text   CR,LF,X-OFF
```

To send multisegment messages to IMS, use the following format:

- Begin the message with a transaction code.
- Terminate each segment with the CR, LF, and X-OFF characters.
- Terminate the message with a period (**.**), and the CR, LF, and X-OFF characters.

Example:

```
transaction_code  segment_1  CR,LF,X-OFF
                  segment_2  CR,LF,X-OFF
                  segment_3  CR,LF,X-OFF
                      . . .
                  segment_n  CR,LF,X-OFF
                             .,CR,LF,X-OFF
```

If you enter X-OFF from paper tape, follow it with at least three RO (rub-out) characters.

IMS sends responses to entered messages immediately only if you enter a message consisting of only an X-OFF character. Otherwise, the terminal remains in input mode and IMS sends no output until 28 seconds have passed without any input’s being received.

Disconnecting from IMS

To terminate a call to IMS, enter an `/IAM DONE` command or press the EOT (end-of-transmission) key after you receive all output.

To terminate while you are sending data, press the EOT key. This causes IMS to ignore the message being entered.

To terminate while output is being sent, repeatedly press the Break key or press the Clear key to disconnect the line. IMS saves all output not sent, unless the terminal is signed on as the inquiry-logical terminal; in this case, IMS discards output.

System Console

IMS communicates with the MVS system console using MVS write-to-operator (WTO) and write-to-operator-with-reply (WTOR) messages.

Enter messages from the console in WTO or WTOR format. MVS passes only the data portion of the reply to IMS. Each reply constitutes one segment of an input message. Each segment can contain either uppercase or lowercase characters. Multisegment messages must end with a period to identify end-of-message. A period can appear either at the end of the last segment (before the end quote), or as a segment itself. Because of WTO and WTOR restrictions, IMS translates any embedded carriage return (CR) characters in output messages to blanks and ignores them.

The following data formats are acceptable. IMS WTO and WTOR reply formats vary depending on the MVS or OS/390 system and the level of JES being used.

Command Line	R *nn*,'/ERESTART CHKPT 0 '
Second Line	R *nn*,'BUILDQ FORMAT RS.'
Transaction Line	R *nn*,'FIRST SEGMENT'
Second Line	R *nn*,'LAST SEGMENT.'
Message Switch	R *nn*,'DESTNAME TEXT'
Second Line	R *nn*,'TEXT 2ND LINE.'

IMS does not print output messages enqueued for the system console on the terminal until you enter a message. This ensures that you are present when output messages are printed. If you do not need to enter an input message, enter a blank or period as a single-character message to cause enqueued output to be printed without input-message processing.

Example	Explanation
27 DFS996I *IMS READY ⋮	IMS is ready for an output segment.
R 27,’.’	Reply to cause enqueued output to be printed.
DFS000I WTOR OPERATOR ARE YOU THERE?	Message 1 from output queue.

Example	Explanation

`DFS000I PLEASE START TRANS`
`XYZ...OK?`
	Message 2 from output queue.

`28 DFS996I *IMS READY` ⋮	IMS is ready for next input segment.

Local Card Reader

IMS supports a local card reader (for example, 2540 or 2502) as an input device. Before operation, however, you must enter a `/START LINE` command from the master terminal to activate the card reader. You can then place the card deck in the hopper and make the reader ready. The reader remains active and able to read cards until you enter a `/STOP LINE` command or until an I/O error occurs.

If you press the end-of-file button, IMS enqueues the last message read for processing regardless of whether it was terminated by a slash-period (/.) delimiter. IMS then issues a read command to allow you to add another deck of cards when needed. If you do not press the end-of-file button, IMS does not enqueue the last message for processing unless it is a single-card input or IMS reads a slash-period. This allows you to fill the hopper with a partial tray of cards without determining whether the last card is also the end of the IMS message. In this case also, a read is left outstanding so you can continue with the input deck by filling the hopper and making the reader ready.

Communicating with IMS

IMS delineates a segment by the presence of an end-of-media (EM) character (X'19', card punch 11-1-8-9) in any column of the input card. IMS ignores any text following the EM character. If no EM character occurs, IMS assumes a segment to be 80 characters in length. The character string slash-period (/.) immediately **preceding** the EM character (if present) or as the first two characters in the following input card terminates a message. This allows you to input variable-length data, even though BSAM is reading fixed-length records.

When you assign a SYSIN data stream to a local reader line, IMS reads the entire file of data and closes the device. You can restart the line using a `/START` command after IMS notifies you that it has processed the file.

Responding to Errors

If an I/O error occurs, follow standard I/O error-recovery procedures for the device. If IMS detects invalid data, it stops the line. Correct the condition, reenter the deck beginning with the transaction that caused the failure, and restart the line from the master terminal. You should use the `/SET` command to identify the input destination in order to prevent data errors in the middle of input streams.

Local Printer

IMS supports a local printer (for example, 1443 or 1403) as a high-speed online output terminal. After you enter a `/START LINE` command from the master terminal, the printer prints output as received from the application program without your intervention. If a permanent I/O error occurs, IMS stops the terminal. Correct the error and restart the line from the master terminal using the `/START LINE PTERM` or `/RSTART LINE PTERM` command.

Magnetic Tape

If you use a tape for SYSIN or SYSOUT data sets, you must change tapes when MVS notifies you of end-of-volume (EOV). By issuing a `/STOP LINE PTERM` command from the master terminal, you can remove a tape before EOV for offline processing. Remove the tape, mount a new one, and enter a `/START LINE PTERM` or `/RSTART LINE PTERM` command to continue processing.

Disk Data Sets

You can allocate a disk data set for use as an intermediate data file accessed by an IMS application program or any other user program. Operation is the same as for a tape file, but IMS provides no means of dismounting disk volumes.

When you allocate a group of disk data sets for the spooled SYSOUT option, the IMS Spool SYSOUT Print utility (DFSUPRT0) can copy one or more of these data sets to SYSOUT. IMS can automatically schedule this utility at end-of-volume on one of the data sets. An end-of-volume condition is also simulated:

- If an I/O error occurs while writing to a spool data set.
- During system initialization, if a spool data set was in use when the system was last running and the spool data set was not printed.

  In this case, you must issue a `/START LINE` command to complete the scheduling procedure.

  Recommendation: Define at least two data sets for your spool line.

You can reschedule the utility with the `/STOP LINE PTERM` command. You can also start the utility at any time by issuing a `/START REGION` *procname* command from the master terminal. IMS generates line numbers and PTERMs in the sequence of the input definition statements during IMS system definition. After the system is generated and ready to run, you can display line numbers to determine what line number was generated for a spool data set.

You must establish your own conventions for controlling special forms. One possible method is to allocate a logical terminal for each report. After you set up the forms for a specific report, lock the other logical terminals using the `/LOCK LTERM` command, then place the terminal in exclusive mode using the `/EXCLUSIVE` command. Thereafter, IMS prints only the data enqueued for the report you want. If you do not use a technique like this one, messages alternate among the logical terminals associated with the line.

Programmable Remote Systems

Operators of programmable remote systems do not communicate directly with IMS, but rather with a user-written program that resides in the remote system's storage. If you use programmable remote systems, you must provide procedures for terminal operators.

Related Reading: *IMS Version 9: Administration Guide: System* and *IMS Version 9: Customization Guide* contain information on communication protocols and programming for the System/3 and the System/7. For specifications for the 3600, FINANCE, type-P secondary logical units, and ISC logical units, see *IMS Version 9: Administration Guide: Transaction Manager*. For a list of remote programmable systems, see *IMS Version 9: Release Planning Guide*.

SLU-1 Devices

For a complete list of the SLU-1 devices that IMS supports, see *IMS Version 9: Release Planning Guide*. This section describes the following SLU-1 devices as examples:

- IBM SLU-1 Communication Terminal.
- IBM SLU-1 Data Communication System (Programmable Models). The SLU-1 supports input from a transmit data set (TDS) or a user data set (UDS), in addition to print data set (PDS) and UDS output if you define them to IMS as unattended.
- IBM SLU-1 Data Communication System (Nonprogrammable Models).

MFS is optionally available for TDS input and PDS output. You can define a SLU-1 device for either attended or unattended operation.

Printers Attached to IMS as SCS SLU-1 Devices

Most of the printers have Cancel, PA1, and PA2 keys that can be used when the device is operating in SNA character string (SCS) mode and is attached to an appropriate controller, such as the 3174, 3274, 3601, or 4701. When these keys are pressed while IMS is printing a message, the following results occur:

- If the Cancel key is pressed, IMS stops transmission and discards the current output message (but does not dequeue it).
- If a PA key is pressed, a string of characters is sent to IMS. You can define this string of characters as transaction code APAK 01 for PA1 and APAK 02 for PA2, or you can use the appropriate IMS user exit routine to modify this string, for example to set it to a command or message switch.

Restriction: You must specify CONSOLE instead of PRINTER1 for the COMPT1 keyword of the TERMINAL macro (or the logon descriptor for the printer). In the TYPE macro or logon descriptor, specify the printer as UNITYPE=SLUTYPE1.

SLU-1 Communication Terminal

This section describes IMS operations for a SLU-1 Communication Terminal or a SLU-1 Data Communication System (nonprogrammable models).

Related Reading: For information on the general features and operation of a SLU-1 device, see *Operating Procedures Guide: IBM 3774 and 3775 Communication Terminals*.

Connecting to IMS Using SLU-1 Devices

Prepare the SLU-1 device for communication before attempting to log on. See the SLU-1 operator's guide for the proper switch settings.

You should log on to terminals that are connected with a switched line using the procedure below. Log on to terminals connected through a nonswitched line using the same procedure, but start with step 2.

1. Establish connection with the computer through a telephone data set.
2. Press the Sys Req key. The Proceed light should go on.
3. Enter the logon message. For example, to log on to an IMS system named IMSA:
   - Enter: `LOGON applid (IMSA)`
   - Response: `DFS3650I SESSION STATUS`

If you want host-initiated logon for a SLU-1 Data Communication System, you must leave the terminal in a line-to-console job. To prepare the SLU-1 terminal for host-initiated logon:

1. Press the Start Job key. For a 3774 or 3775, also press the S key.
2. Press the Line key.
3. Press the Console key.
4. Press the EOM (end of message) key.

Communicating with IMS Using SLU-1 Devices

You can begin communication with IMS after you complete the logon procedure. If an output message is enqueued for the terminal, IMS attempts to send it immediately after the `DFS3650I` message. If you begin to enter an input message before you receive the output message, IMS sends the output message later. Methods of stopping an output message in progress are described in “Receiving an Output Message.”

Entering Single-Segment Input: Enter single-segment input as follows:

1. Enter the input data.
2. Press the carriage-return key. This step is optional.
3. Press the EOM key.

Entering Multisegment Input: **Recommendation:** For a SLU-1 Communication Terminal, set the AUTO switch to OFF when you enter multisegment input. When you set the AUTO switch to AUTO, you send data in the buffer plus an EOM to IMS when you press the carriage-return key or the Form Feed key.

Enter multisegment input as follows:

1. Enter the first segment of the input.
2. Press the carriage-return key.
3. Press the EOB (end of buffer) key to send the buffer contents, if required. If you want to enter another segment in this buffer, do not press the EOB key.
4. Enter the next input segment.
5. Press the carriage-return key.
6. If there are more segments to enter, return to step 3. If not, press the EOM key.

Receiving an Output Message: IMS sends all output messages for a SLU-1 device in their entirety unless you interrupt them at the device using the Cancel and Attn keys as follows:

- The Cancel key stops the current message, which IMS then discards. If another output message is presently enqueued, IMS sends it.
- The Attn key causes IMS to finish sending the current message. When the message completes, you can enter an input message.
- The Attn key followed by the Cancel key stops the current message, which IMS then discards. When the message is stopped, you can enter an input message.

SLU 1 with MFS: During IMS system definition, you can define a SLU-1 terminal to work with MFS. Like a 3270 terminal, system-supplied formats are available. Even when defined to operate with MFS, the SLU-1 terminal operates in unformatted mode (using IMS basic edit) until one of the following occurs:

- You enter `//`*`midname`*.

- MFS processes an output message to the terminal using an MFS message output descriptor (MOD) that names an MFS message input descriptor (MID) to be used to process subsequent input data.

If you enter data on the same line as //*midname*, separate //*midname* from the data with a blank. IMS discards the //*midname* and the blank, and formats the data according to the named MID. If you enter //*midname* without any data on the same line, IMS considers the next line received from the terminal to be the first line of the message.

When IMS processes an output message to a SLU-1 terminal using a MOD that names a MID, MFS uses that MID to format the next input from that terminal. An application program, the IMS /FORMAT command, a message switch, or some IMS functions can create this output message.

When in “formatted mode” (using MFS), a SLU-1 terminal remains in formatted mode until one of the following occurs:

- You enter two slashes (//) or two slashes followed by a blank. The terminal returns to unformatted mode, and IMS discards the two blanks. The two slashes are escape characters.
- You enter two slashes, a blank, and data. MFS returns the terminal to unformatted mode, IMS discards the slashes and the blank, and formats the data using basic edit.
- IMS sends an output message that does not name a MOD to the terminal.
- IMS sends an output message whose MOD does not name a MID to the terminal.

IMS uses MFS default formats for many output messages. When a terminal receives such a message, the terminal goes into formatted mode, and MFS processes the next input message unless you use the escape characters. If in doubt, use the escape characters. See *IMS Version 9: Administration Guide: Transaction Manager* for information about defining SLU-1 terminals to operate with MFS.

Response Mode: During IMS system definition or in the ETO logon descriptor, you can specify that a SLU-1 terminal attached to IMS should operate in forced-response mode, nonresponse mode, or transaction-dependent response mode.

Disconnecting SLU-1 Devices from IMS

To disconnect from IMS, press the Sys Req key and then enter a logoff command. For a SLU-1 Data Communication System, you can also press the Stop Job key.

SLU-2 Devices

See *IMS Version 9: Release Planning Guide* for a complete list of the SLU-2 devices that IMS supports. The 3290 Information Panel SLU-2 device is described in “3290 Information Panel” on page 377 as an example.

Connecting to IMS Using SLU-2 Devices

When you connect to or disconnect from IMS using a SLU-2 device, the procedure is the same regardless of the device. A SLU-2 device can initiate a session with IMS in several ways:

- The VTAM network operator can enter the `VARY NET` command (`VARY NET,ID=a,LOGON=b`).
- You can enter the `/OPNDST` command from the IMS master terminal.
- VTAM can make the unit automatically available if it is defined in the VTAM definition as belonging to IMS.

If you define the terminal as the master terminal, IMS automatically enables the terminal for a session.

Disconnecting SLU-2 Devices from IMS

Close a terminal session by issuing a `/RCLSDST` command.

3290 Information Panel

When communicating with IMS using a 3290, the terminal operator might need to know that how the PA1 key operates is based on what option is selected for presenting pages to a partition. These options and the resulting use of the PA1 key are described in *IMS Version 9: Administration Guide: Transaction Manager*.

If an error occurs, the 3290 session usually terminates. The 3290, not IMS, handles device errors.

Related Reading: For information on the general features and operation of the 3290 Information Panel, see *3290 Information Panel Description and Reference*. Additional information on partitioning is in *IMS Version 9: Application Programming: Transaction Manager*.

NTO Logical Units

IMS allows start-stop devices such as TTY, 2740-1, 2741, and TTY-compatible devices (for example, the 3101) to attach to IMS through VTAM. The Network Terminal Option (NTO) Program Product handles the device characteristics associated with line control and communication protocols, and translates them into SNA-SLU 1 communication protocols handled by VTAM and IMS. The operational characteristics of each device do not change, but session initiation and termination are modified as described below.

For a complete list of the NTO devices that IMS supports, see *IMS Version 9: Release Planning Guide*.

Connecting to IMS Using NTO Devices

Before you can initiate an NTO session, IMS must recognize the NTO device. For this to occur, you must do both of the following:

- Define the NTO device to IMS during IMS system definition. Identify the device using the `PU` keyword in the TERMINAL macro. This keyword specifies the actual hardware that this NTO device represents. For more information on the `PU` keyword, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.
- Enter a `/START DC` command from the IMS master terminal.

IMS initiates a session with an NTO device when one of the following occurs:

- You enter the `LOGON APPLID(nnnnnnnn)` command at the NTO terminal.
- You enter an `/OPNDST` command (for nonswitched devices only) from the IMS master terminal.

- The VTAM network operator issues a VARY command with the LOGON= keyword (for nonswitched devices only).

The NTO device receives a DFS3650I message when session initiation has completed successfully.

Communicating with IMS Using NTO Devices

This section describes how to communicate with IMS from an NTO device.

Control Characters

IMS uses the following control characters to delineate messages and message segments:

. (period)
End-of-message. This character signifies the end of a multisegment message when you do not use MFS.

CR
Carriage return (X'0D').

LF
Line feed (X'25').

Single-Segment Messages

To send data to IMS, use the following format:

- Begin the message with a transaction code.
- End the message with the CR and LF characters.

Example:

```
transaction_code  text   CR,LF
```

Multisegment Messages

To send data to IMS, use the following format:

- Begin the message with a transaction code.
- End each segment with the CR and LF characters.
- End the message with a period (**.**), followed by the CR and LF characters.

Example:

```
transaction_code  segment_1  CR,LF
                  segment_2  CR,LF
                   . . .
                  segment_n  CR,LF
                             .,CR,LF
```

You can reverse the order of the CR and LF characters.

Operation of the NTO device is the same as for those devices without NTO and VTAM. For additional information on entering messages, receiving messages, using response mode, using MFS, and using IMS commands from the terminal, see the preceding sections on each type of physical device.

Restriction: The /IAM command is not supported for NTO devices.

Disconnecting NTO Devices from IMS

The session between an NTO device and IMS terminates when any of the following occurs:

- IMS terminates.
- VTAM terminates.

- The network operator enters a `VARY INACT` command.
- You do one of the following at the NTO terminal:
  - Issue a `/RCLSDST` command.
  - Enter a logoff command.
  - Disconnect the phone connection for a switched device.
- You enter the `/CLSDST` command from the IMS master terminal.
- Data or transmission errors force termination of the session.

When an NTO session fails, you must reconnect it as described in “Connecting to IMS Using NTO Devices” on page 377. For information on responding to errors, see the component manuals for each device type and *Network Terminal Option Installation Releases 1, 2, 2.1, 3*.

Restriction: You must not stop the NTO logical unit using the IMS `/STOP` command. The network operator must issue the VTAM `VARY` command (with `ANS=ON` for switched devices).

Special Operational Notes and Restrictions

For TTY devices, IMS system messages cause a carriage return (CR) and linefeed (LF) before and after the message. Other output does not cause this to happen.

For 3101 devices acting as TTY devices, IMS sends a carriage return (CR) as X'0D'. This has the same effect on the 3101 screen for IMS system messages as the carriage return (CR) and linefeed (LF) do for the TTY.

There are no component attachments to an NTO device. You must define each component as a unique NTO device to VTAM, the NTO Program Product, the NCP, and IMS.

An NTO device cannot be used as an IMS master terminal.

Master Terminal

You can define any of the following devices as the master terminal for an IMS subsystem:

- System console
- IBM 2740 Communication Terminal (with station-control feature and nonswitched)
- IBM 3270 Information Display System (nonswitched)
- SLU-1 device (attended mode only)
- SLU-2 device

If you use a 3270 device, IMS requires that you use a 3270 display station with a 3270 printer for the master terminal (IMS does not allow the 3284 Model 3 printer). The master terminal cannot be an ETO terminal or an LU 6.2 device.

Appendix B. IMS Type-1 and Type-2 Command Equivalents

Table 66 shows the different instances of the IMS type-1 commands and type-2 commands that perform similar actions.

Table 66. IMS Type-1 and Type-2 Command Equivalents

Task	Type-1 Command	Type-2 Command
Change the value for the current priority of a transaction.	/ASSIGN CPRI new_current_priority TO TRAN name	UPDATE TRAN NAME(name) SET(CPRI(new_current_priority))
Change the value for the limit count of a transaction.	/ASSIGN LCT new_lmct_number TO TRAN name	UPDATE TRAN NAME (name) SET(LCT(new_limit_count))
Change the value for the limit priority of a transaction.	/ASSIGN LPRI new_lpri_number TO TRAN name	UPDATE TRAN NAME(name) SET(LPRI(new_limit_priority))
Change the value for the normal priority of a transaction.	/ASSIGN NPRI new_npri_number TO TRAN name	UPDATE TRAN NAME(name) SET(NPRI(new_normal_priority))
Change the value for the parallel processing limit count of a transaction.	/ASSIGN PARLIM new_parlim_number TO TRAN name	UPDATE TRAN NAME(name) SET(PARLIM(new_parallel_limit))
Change the value for the processing limit count of a transaction.	/ASSIGN PLCT new_plmct_number TO TRAN name	UPDATE TRAN NAME(name) SET(PLCT(new_processing_limit))
Change the limit on the number of application program output segments allowed in message queues for each GU call.	/ASSIGN SEGNO new_segno_number TO TRAN name	UPDATE TRAN NAME(name) SET(SEGNO(new_segment_number))
Create or change the limit on the size of application program output segments allowed in message queues for each GU call.	/ASSIGN SEGSZ new_segsize_number TO TRAN name	UPDATE TRAN NAME(name) SET(SEGSZ(new_segment_size))
Change the class number of a transaction.	/ASSIGN TRAN name TO CLS new_class_number	UPDATE TRAN NAME(name) SET(CLASS(new_class_number))
Stop updates to a database.	/DBDUMP DB dbname	UPDATE DB NAME(name) STOP(UPDATES) OPTION(FEOV)[1]
Stop the accessing and updating of an area.	/DBRECOVERY AREA areaname	UPDATE AREA NAME(name) STOP(ACCESS)
Stop the accessing and updating of all areas and databases of the data group.	/DBRECOVERY DATAGRP datagrpname	UPDATE DATAGRP NAME(name) STOP(ACCESS)
Stop access to the database and take it offline.	/DBRECOVERY DB dbname	UPDATE DB NAME(name) STOP(ACCESS) OPTION(FEOV)[1]
Display information about an area.	/DISPLAY AREA	QUERY AREA
Display the status of a database.	/DISPLAY DB dbname...dbnamen | ALL	QUERY DB
Display information about databases with the specified status.	/DISPLAY STATUS DB	QUERY DB STATUS
Display information about transactions with the specified status.	/DISPLAY STATUS TRANSACTION	QUERY TRAN NAME(tranname) STATUS (IOPREV,LCK,QERR,SUSPEND, STOQ,STOSCHD,USTO)
Display information about a transaction.	/DISPLAY TRAN tranname	QUERY TRAN NAME(tranname) SHOW(ALL)

Table 66. IMS Type-1 and Type-2 Command Equivalents (continued)

Task	Type-1 Command	Type-2 Command
Display all of the transactions.	/DISPLAY TRAN ALL	QUERY TRAN SHOW(ALL)
Display all of the transactions on the shared queues with a global queue count.	/DISPLAY TRAN tranname QCNT	QUERY TRAN NAME(tranname) SHOW(QCNT)
Stop the use of a database.	/LOCK DB dbname	UPDATE DB NAME(name) SET(LOCK(ON))
Stop the scheduling of transactions.	/PSTOP TRAN name	UPDATE TRAN(name) START(Q) STOP(SCHD)
Stop input messages for a particular transaction code.	/PURGE TRAN name	UPDATE TRAN NAME(name) START(SCHD) STOP(Q)
Start the area.	/START AREA areaname	UPDATE AREA NAME(name) START(ACCESS)
Start the data group.	/START DATAGRP datagrpname	UPDATE DATAGRP NAME(name) START(ACCESS)
Start a database and change access intent of the database.	/START DB ACCESS	UPDATE DB START(ACCESS) SET(ACCTYPE())
Start a database.	/START DB dbname	UPDATE DB NAME(name) START(ACCESS)
Start a transaction.	/START TRAN	UPDATE TRAN NAME(name) START(Q,SCHD,SUSPEND)
Stop an area.	/STOP AREA areaname	UPDATE AREA NAME(name) STOP(SCHD)
Stop a data group.	/STOP DATAGRP datagrpname	UPDATE DATAGRP NAME(name) STOP(SCHD)
Stop a database.	/STOP DB dbname	UPDATE DB NAME(name) STOP(SCHD)
Stop the queuing and scheduling of messages destined for a transaction.	/STOP TRAN name	UPDATE TRAN NAME(name) STOP(Q,SCHD)
Starts the tracing of a transaction.	/TRACE SET ON TRAN name	UPDATE TRAN NAME(name) START(TRA)
Stops the tracing of a transaction.	/TRACE SET OFF TRAN name	UPDATE TRAN NAME(name) STOP(TRA)
Unlock a database.	/UNLOCK DB dbname	UPDATE DB NAME(name) SET(LOCK(OFF))

[1] Note that these commands do not automatically issue checkpoints unless `OPTION(FEOV)` is specified.

Related Reading: For detailed information about these commands, see *IMS Version 9: Command Reference*.

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 give 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.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

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.

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.

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

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

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

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

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 measurement 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 is for planning purposes only. The information herein is subject to change before the products described become available.

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:

This information contains sample application programs in source language, which illustrates 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. 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 IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

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

BookManager
CICS
CUA
DataPropagator
DB2 UDB for z/OS
DFS
IBM
IMS
IMS/ESA
Language Environment
MVS/ESA
NetView
OS/390
RACF
SAA
VTAM
z/OS

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc., in the United States, other countries, or both.

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

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

Product Names

In this book, the following licensed programs have shortened names:

- IBM DB2 Universal Database for z/OS is referred to as “DB2 UDB for z/OS”.
- Enterprise COBOL for z/OS and OS/390 is referred to as “COBOL”.
- PL/I for MVS & VS is referred to as “PL/I”.

Bibliography

This bibliography includes all the publications cited in this book, including the publications in the IMS library

- *VTAM V4R2 for MVS/ESA, VM/ESA, VSE/ESA Messages and Codes*, SC31-6493
- *CICS-Supplied Transactions*, SC33-1168
- *CICS Customization Guide*, SC34-5989
- *CPI Communications Reference*, SC26-4399
- *IMS Queue Control Facility for z/OS*, SC26-9685
- *MVS Programming: Sysplex Services Guide*, SA22-7617
- *z/OS MVS System Commands*, SA22-7627
- *Tivoli NetView for z/OS Installation: Getting Started*, SC31-8872
- *Tivoli NetView for z/OS User's Guide*, SC31-8849
- *Network Terminal Option Installation Releases 1, 2, 2.1, 3*, SC38-0298
- *MVS Initialization and Tuning Reference*, SA22-7592
- *MVS Initialization and Tuning Guide*, SA22-7591
- *Operating Procedures Guide: 3774 and 3775 Communication Terminals*, GA27-3094
- *SAA CPI Resource Recovery Reference*, SC31-6821
- *3270 Information Display System, Configurator*, GA27-2849
- *3270 IDS Operator's Guide 3275*, GA27-2742
- *3290 Information Panel Description and Reference*, GA23-0021

IMS Version 9 Library

ZES1-2330	ADB	*IMS Version 9: Administration Guide: Database Manager*
ZES1-2331	AS	*IMS Version 9: Administration Guide: System*
ZES1-2332	ATM	*IMS Version 9: Administration Guide: Transaction Manager*
ZES1-2333	APDB	*IMS Version 9: Application Programming: Database Manager*
ZES1-2334	APDG	*IMS Version 9: Application Programming: Design Guide*
ZES1-2335	APCICS	*IMS Version 9: Application Programming: EXEC DLI Commands for CICS and IMS*
ZES1-2336	APTM	*IMS Version 9: Application Programming: Transaction Manager*
ZES1-2337	BPE	*IMS Version 9: Base Primitive Environment Guide and Reference*
ZES1-2338	CR	*IMS Version 9: Command Reference*
ZES1-2339	CQS	*IMS Version 9: Common Queue Server Guide and Reference*
ZES1-2340	CSL	*IMS Version 9: Common Service Layer Guide and Reference*
ZES1-2341	CG	*IMS Version 9: Customization Guide*
ZES1-2342	DBRC	*IMS Version 9: DBRC Guide and Reference*
ZES1-2343	DGR	*IMS Version 9: Diagnosis Guide and Reference*
ZES1-2344	FAST	*IMS Version 9: Failure Analysis Structure Tables (FAST) for Dump Analysis*
ZES1-2346	OLR	*IMS Version 9: HALDB Online Reorganization Guide and Reference*
ZES1-2347	JGR	*IMS Version 9: IMS Java Guide and Reference*
ZES1-2348	IIV	*IMS Version 9: Installation Volume 1: Installation Verification*
ZES1-2349	ISDT	*IMS Version 9: Installation Volume 2: System Definition and Tailoring*
ZES1-2350	INTRO	*IMS Version 9: An Introduction to IMS*
ZES1-2351	MIG	*IMS Version 9: Master Index and Glossary*
ZES1-2352	MC1	*IMS Version 9: Messages and Codes, Volume 1*
ZES1-2353	MC2	*IMS Version 9: Messages and Codes, Volume 2*
ZES1-2354	OTMA	*IMS Version 9: Open Transaction Manager Access Guide and Reference*
ZES1-2355	OG	*IMS Version 9: Operations Guide*
GC17-7831	RPG	*IMS Version 9: Release Planning Guide*

ZES1-2358	URDBTM	*IMS Version 9: Utilities Reference: Database and Transaction Manager*
ZES1-2359	URS	*IMS Version 9: Utilities Reference: System*

Supplementary Publications

GC17-7825	LPS	*IMS Version 9: Licensed Program Specifications*
ZES1-2357	SOC	*IMS Version 9: Summary of Operator Commands*

Publication Collections

LK3T-7213	CD	IMS Version 9 Softcopy Library
LK3T-7144	CD	IMS Favorites
LBOF-7789	Hardcopy and CD	Licensed Bill of Forms (LBOF): IMS Version 9 Hardcopy and Softcopy Library
SBOF-7790	Hardcopy	Unlicensed Bill of Forms (SBOF): IMS Version 9 Unlicensed Hardcopy Library
SK2T-6700	CD	OS/390 Collection
SK3T-4270	CD	z/OS Software Products Collection
SK3T-4271	DVD	z/OS and Software Products DVD Collection

Accessibility Titles Cited in this Book

SA22-7787		z/OS V1R1.0 TSO Primer
SA22-7794		z/OS V1R1.0 TSO/E User’s Guide
SC34-4822		z/OS V1R1.0 ISPF User’s Guide, Volume 1

Index

Program Number: 5655-J38

IBM Confidential
Printed in USA

ZES1-2355-00

Spine information:

IBM IMS Operations Guide Version 9