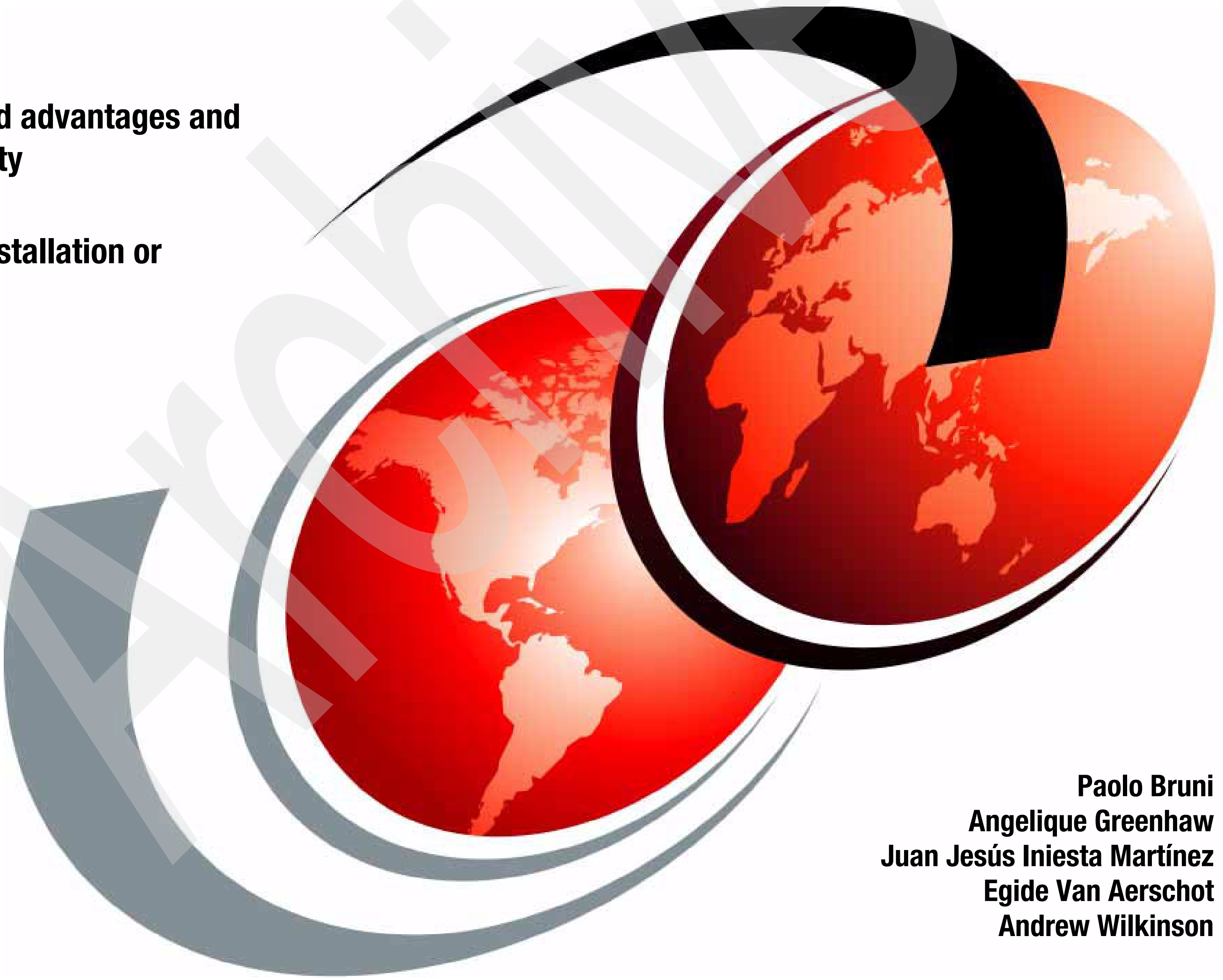
Information Management

IMS Version 11 Technical Overview

Explore IMS 11 functions

Understand advantages and applicability

Plan for installation or migration



Paolo Bruni
Angelique Greenhaw
Juan Jesús Iniesta Martínez
Egide Van Aerschot
Andrew Wilkinson

Redbooks

ibm.com/redbooks

International Technical Support Organization

IMS Version 11 Technical Overview

October 2009

SG24-7807-00

Note: Before using this information and the product it supports, read the information in "Notices" on page xxi.

First Edition (October 2009)

This edition applies to Version 11 of Information Management System (IMS) Transaction and Database Servers (program number 5635-A02) and IMS Enterprise Suite for z/OS, Version 1.1 (program numbers 5655-T60 and 5655-T61).

Note: This book is based on a pre-GA version of a product and may not apply when the product becomes generally available. We recommend that you consult the product documentation or follow-on versions of this book for more current information.

Contents

Tables

Figures

Examples

Notices

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at `http://www.ibm.com/legal/copytrade.shtml`

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®
BookManager®
CICS®
DB2 Connect™
DB2 Universal Database™
DB2®
Distributed Relational Database Architecture™
DRDA®
Enterprise Storage Server®
Enterprise Workload Manager™
FlashCopy®
IBM®
IMS™
Language Environment®
MQSeries®
OS/390®
Parallel Sysplex®
ProductPac®
RACF®
Rational®
Redbooks®
Redpaper™
Redbooks (logo) ®
RETAIN®
S/390®
SecureWay™
System z10™
System z9®
System z®
SystemPac®
VTAM®
WebSphere®
z/Architecture®
z/OS®
z9®

The following terms are trademarks of other companies:

Snapshot, and the NetApp logo are trademarks or registered trademarks of NetApp, Inc. in the U.S. and other countries.

Red Hat, and the Shadowman logo are trademarks or registered trademarks of Red Hat, Inc. in the U.S. and other countries.

Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

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

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

Preface

IMS™ provides leadership in performance, reliability and security to help you implement the most strategic and critical enterprise applications. IMS also keeps pace with the evolving IT industry.

Information Management System Version 11 (IMS 11) provides an open, integrated, and distributed data access solution; improves connectivity and ease of use; increases system availability; and offers an architectural road map that supports future growth.

With IMS 11, a suite of Universal drivers supports IMS database programmatic access with Type-4 and Type-2 connectivity. You can now access IMS in a uniform way using the industry standard Distributed Relational Database Architecture™ (DRDA®) protocol from any platform and from within the most strategic run times, opening growth and expansion opportunity. IMS DB metadata is exposed with the standard JDBC API and therefore can be consumed and visualized by JDBC tooling. Query syntax uses standard query language syntax.

The new IMS Enterprise Suite V1.1 components are designed to enhance your use of IMS applications and data by delivering innovative capabilities that enhance connectivity, expand application development, extend standards and tools for a service-oriented architecture (SOA), ease installation, and provide simplified interfaces.

IMS 11 also helps lower your IT costs in the areas of business flexibility, simplified administration, and growth.

In this IBM® Redbooks® publication, we explore the new features of IMS 11. We review the available material and include feedback from early users. The intent is to highlight the major new functions and facilitate installation and migration.

The team who wrote this book

This book was produced by a team of specialists from around the world working at Silicon Valley Laboratory, San Jose, California.

Paolo Bruni is an Information Management software Project Leader at the International Technical Support Organization based in Silicon Valley Lab, San Jose. He has been an IMS developer, advocate, and introduction program manager and has recently come back to IMS after a few decades working mostly on DB2®. During Paolo's many years with IBM, both in development and in the field, his work has been mostly related to database systems.

Angelique Greenhaw joined IBM in 2000 after she graduated from Arizona State University with a Bachelors degree in Computer Information Systems. She spent six months in IMS test before moving to IMS Systems development, where she specialized in the Online Change function for the next six years. Angie contributed to new IMS functionality and also worked as a Level 3 Service Representative in the Online Change component area. She also spent three years as the IMS Development Representative at SHARE, a technical conference for IBM product users. In January 2007, Angie joined IMS Advanced Technical Support as an IT Specialist, where she is a primary resource in the areas of IMS Dynamic Resource Definition and Online Change. She co-authored *IBM IMS Version 10 Implementation Guide*, SG24-7526.

Juan Jesús Iniesta Martínez is a Certified Consulting I/T Specialist with IBM Global Services in Spain. He has 22 years of experience in the IMS field. He has the IBM IMS Certificates of Achievement of Mastery in Fundamentals, Systems Programming, and Database Administrator. Prior to joining IBM in 1995, he worked for 8 years as an IMS Systems Programmer and Technical Support Specialist in a large Spanish bank. Since Juan joined IBM, he has been assigned to several important financial entities in Spain, working in IMS projects, and providing IMS support and guidance. Since 2003, Juan manages the Spanish IMS GUIDE (GSE) users working group. His main areas of expertise and responsibilities in the IMS field include installation and maintenance, problem determination and related products, Parallel Sysplex®, data sharing, IMS shared queues, and IMSPlex installations. Juan co-authored *IMS in the Parallel Sysplex, Volume I: Reviewing the IMSplex Technology,* SG24-6908, *Volume II: Planning the IMSplex*, SG24-6928, and *Volume III: IMSplex Implementation and Operations,* SG24-6929.

Egide Van Aerschot holds an Engineering degree in Electricity and Nuclear Physics from the University of Leuven, Belgium. He joined IBM in 1967 and was responsible for many computer installations related to tele-processing and database management in Belgium. In 1997 he moved from IBM Belgium to IBM France, where he works as an Architect and Consultant at the IBM Program Support Center in Montpellier. Since 1997, he has specialized in Java™, service-oriented architecture, IMS and WebSphere® applications, mainly on z/OS® systems, and has participated in many projects related to the Internet. Egide is co-owner of the patent "Methods, systems, program product for transferring program code between computer processes." Currently, Egide is a contractor for the Zinteg C.V. He teaches for IBM, System z® WebSphere Application Server, and WebSphere MQ classes in Northern Europe.

Andrew Wilkinson is a Technical Specialist for IBM Software Group in the UK. He holds a degree in Natural Philosophy from Glasgow University. He has 28 years of experience with IMS. Andrew joined IBM in 1981 as a PL/I programmer writing IMS DB/DC applications. In 1990 Andrew became an IMS Systems Programmer, working with IBM IMS systems and then those of companies outsourced to IBM. Since 2002, Andrew has been in IBM Software Group helping IBM customers get the most out of IMS and IBM IMS tools.

The authors in SVL. From left to right: Andrew, Paolo, Juan, Angie, and Egide

Special thanks from the authors to the IMS development team of the Silicon Valley Lab (SVL) for their knowledgeable, friendly, and open-minded collaboration.

Thanks to the following people for their contributions to this project:

Rich Conway
Bob Haimowitz
Emma Jacobs
Diane Sherman
International Technical Support Organization

Jim Bahls
Tom Bridges
John Butterweck
Himakar Chennapragada
Nate Church
Demetrios Dimato
Sally Gehring
Kevin Hite
Shu Hsu
Jenny Hung
Barbara Klein
Terry Krein
Yee-Rong Lai
Rose Levin
Huong Loi
Neela Mehta
Dean Meltz
Beth Moore
Bruce Naylor
Danny Nguyen
Huan Nguyen
Daniel Reilly
Frank Ricchio
Pat Schroeck
Alan Smith
Maida Snapper
Qiuhong Sun
Stan Y Sun
Richard Tran
Jack Yuan
Mark Ziebarth
IBM Silicon Valley Lab

Alan Cooper
Alison Coughtrie
IBM UK

Ken Blackman
Rich Lewis
Advanced Technical Support, Americas

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

1

IMS 11 at a glance

In this chapter, we outline the enhancements in IMS 11, with references to the places in this book where you can find more details.

Version 11 is a major enhancement to IMS with many new and useful features that improve integration, openness, manageability, and scalability. These enhancements can help you address your On Demand Business needs.

Chief among the enhancements are:

- Open Database, which greatly simplifies access to data in IMS databases from applications that do not run on z/OS.
- Transaction expiration, which allows you to automatically discard messages on the basis of age.

In addition, many smaller enhancements are also very useful.

We discuss the enhancements, grouping them by the following topics:

- Integration
- Openness
- Manageability
- Scalability

If you are migrating from IMS 9, all IMS 10 enhancements are also new to you. Refer to the following locations for information about the IMS 10 enhancements:

- A summary is in 1.5, “Enhancements in IMS 10 if migrating from IMS 9” on page 9.
- A full discussion is in *IBM IMS Version 10 Implementation Guide: A Technical Overview*, SG24-7526.

1.1 Integration

IMS 11 has several features to help you more easily integrate IMS into your enterprise. This section describes those features.

1.1.1 IMS Connect

IMS Connect is at the heart of the IMS integration strategy.

The biggest change is for Open Database. IMS Connect becomes the entry point for DRDA requests for access to IMS databases. This means that IMS Connect becomes important also for database-only users of IMS.

Two Java drivers, named the IMS Universal drivers, provide Open Database Access (ODBA) either directly using ODBA (if running on the same LPAR) or using Distributed Relational Database Architecture (DRDA) with IMS Connect (if not running on the same LPAR). The drivers handle the various protocols so that the programmer does not have to.

In addition, other smaller enhancements to IMS Connect can improve its integration with WebSphere Application Server.

For more information, refer to 6.1, "IMS Connect and new functions" on page 158.

1.1.2 IMS Enterprise Suite

Although not part of IMS 11, the IBM IMS Enterprise Suite is made available for IMS 11 and contains several useful functions. One of the best features about the Enterprise Suite is that it does not require additional charges.

Among other features, the Enterprise Suite contains:

- A new version of the DLIModel utility. This version now supports PL/I structures and has a number of other enhancements.

  Refer to 6.3, "IBM IMS Enterprise Suite DLIModel utility" on page 187 for details.
- IMS Connect APIs. These APIs help programmers more easily communicate directly with IMS Connect, rather than with the IMS TM Resource Adapter or the IMS SOAP Gateway.

  Refer to 6.1.5, "IMS Enterprise Suite V1.1 Connect API for Java and C" on page 178.
- A JMS API for application programs running in Java-dependent region that want to make synchronous callout requests

  Refer to 6.4, "IBM IMS Enterprise Suite JMS API" on page 192 for details.

The Enterprise Suite is summarized in A.2, "IMS Enterprise Suite" on page 308.

1.1.3 IMS Transaction Manager Resource Adapter

The IMS TM Resource Adapter now has support for MFS, enabling you to more easily create business processes in WebSphere Integration Developer from your MFS source. After you have created the processes in WebSphere Integration Developer, you can deploy and run them in WebSphere Process Server.

Refer to "Resource Adapters" on page 196 for details.

1.2 Openness

IMS 11 is more open than previous version of IMS, particularly the Database Manager.

1.2.1 Open Database

The main change introduced by IMS Open Database is the ability to easily access IMS databases from platforms other than z/OS. This access is through TCP/IP by using the DRDA protocol; applications are shielded from the complexity of DRDA by a new set of Java drivers, known as the IMS Universal drivers.

IMS Connect gains a role as the entry point for DRDA requests into IMS. A new Common Service Layer (CSL) address space, the Open Database Manager (ODBM) connects to the control region. Figure 1-1 shows this architecture at a high level.

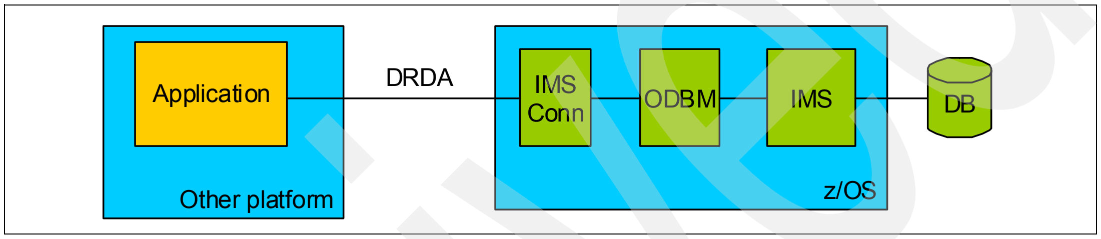


Figure 1-1 Open Database Access to IMS

The diagram does not show that Open Database exploits IMSplex. A DRDA request can arrive in IMS Connect on one LPAR but be serviced by an ODBM on another LPAR. This approach allows the workload, as a whole, to increase up to the capacity of the IMSplex.

Existing ODBA applications (including existing WebSphere Application Server for z/OS applications and DB2 stored procedures) can also take advantage of the ODBM to insulate the IMS control region against application abends.

These functions are described in 3.1, “IMS Open Database” on page 76.

1.2.2 IMS SOAP Gateway

For IMS 11, the existing IMS SOAP Gateway has been extensively enhanced and packaged as part of the new IMS Enterprise Suite.

IMS SOAP Gateway now supports:

- Multi segment messages
- Security through Web Services Security
- Business Event using:
  - REST (through WebSphere Business Monitor)
  - SOAP (through WebSphere Event Monitor)

For information about the IMS SOAP Gateway, refer to 6.2, “IBM IMS Enterprise Suite SOAP Gateway” on page 180.

1.3 Manageability

IMS 11 can help you more easily manage your system in several ways. Transaction expiration and database quiesce are the main improvements.

1.3.1 Transaction expiration

In earlier versions of IMS, you could have had problems with transactions arriving from another platform when transactions are queuing. The sender of a transaction might have timed out, before the transaction had a chance to run, informing the user that the transaction had failed. However, the transaction would still be queuing in IMS and might later run successfully.

IMS 11 helps avoid this problem by introducing the option to discard a transaction before execution, on the basis of age.

You specify the expiration time on the transaction definition, but you can override this on the message header if the transaction comes through Open Transaction Manager Access (OTMA).

For more information about this topic, refer to 4.5, “Transaction expiration” on page 138.

We discuss the OTMA considerations separately in 4.3.2, “OTMA transaction expiration and input message timeout” on page 115.

1.3.2 Database quiesce

In IMS versions prior to V11, synchronizing the stopping of several databases, especially when data sharing, was difficult. It meant that you typically had to write an Automated Operator Interface (AOI) program to issue the command **`/DBR`** on all the databases in a business application and confirm that they had all been stopped. Often, the next operation would be to start all the databases again, in order to create a recovery point. This process was time-consuming, especially if batch messaging programs (BMPs) were running when the commands were issued.

IMS 11 has an elegant solution for this problem. Refer to 3.2, “Database quiesce” on page 85 for more information.

1.3.3 System manageability

This section discusses several useful improvements that have been made to the IMS system component.

User exits

IMS 11 introduces three user exits:

- An initialization/termination exit (invoked at IMS initialization and termination)
- An IMS Common Queue Server (CQS) event exit (invoked when IMS receives a CQS event)
- An IMS CQS structure event exit (invoked when IMS receives a CQS structure event)

The ability to refresh certain user exits in flight is also added to IMS 11 through maintenance. This enhancement applies to non-Base Primitive Environment (non-BPE) exits. It is separate from the existing ability to refresh BPE exits.

For more information, refer to 2.1, “User Exit Interface” on page 14.

Dump formatting

In IMS 11, an online dump formatting exit, DFSAFMX0, replaces DFSAFMD0 and is easier to install because it does not have to reside in LPALIB.

Refer to 2.8, “Dynamic Abend Dump Formatting exit” on page 70.

The IMS 11 Interactive Dump Formatter has a new feature that can write records to a log file from data in the log buffers in the dump. This approach means you can easily create a log file containing only the records pertinent to the problem, without having to look in your system log data sets (SLDSs).

Refer to 2.2, “IMS Interactive Dump Formatter” on page 25.

Installation

IMS 11 extends the Syntax Checker to cover the PROCLIB members introduced for ODBM and BPE-based DBRC. There is also support for new BPE keywords and several usability improvements.

Refer to 2.4, “Syntax Checker” on page 32.

A new IVP for Open Database confirms that your installation is correct. In addition, many other changes are available, especially if you are migrating from IMS 9.

Refer to 2.3, “Installation verification procedure” on page 31 for more details.

Problem diagnosis

In IMS 11, Knowledge-Based Log Analysis (KBLA) panels support scrolling, which makes them easier to use when in split-screen mode. Also KBLA can now allocate multi-volume output data sets.

Refer to 2.6, “KBLA” on page 50.

New keywords and a new option for the **/DIA** (diagnose) command enable you to snap a copy for:

- APPC or OTMA shared queue SCD extension
- Module information
- CQS structure information

Refer to 2.7, “The /DIAGNOSE command” on page 63.

Security

The two changes to security are:

- By default, IMS 11 honors the RACF® setting of mixed-case passwords.

  This change applies to both the control region (VTAM® access) and IMS Connect (TCP/IP access).

  Previous versions of IMS defaulted to uppercase only.

  See 8.1, “Mixed-case password support” on page 286.

- The option of LTERM-based security is changed.

  This change is for customers who do not force users of static terminals to sign on, but still require a user ID for security. If you request this option for a terminal (using the AUTOSIGN option) IMS uses LTERM name as the user ID. This option is also available by PTF in IMS 10 and IMS 9.

  Refer to 8.3, “Automatic signon for static terminals” on page 300.

1.3.4 Database Manager manageability

In addition to the major enhancements, other improvements apply to the IMS database and are described in this section.

Dynamic allocation of ACBLIB

IMS 11 allows you to modify the inactive ACBLIB concatenation while IMS is running. You can:

- Compress ACB libraries.
- Change the concatenation sequence.
- Add or remove ACB libraries.

This improvement is particularly useful in an IMSplex with Global Online Change, because earlier versions of IMS required an IMSplex outage for such changes.

Refer to 3.3.2, “Dynamic allocation of ACBLIBA/B” on page 89 for instructions regarding how to exploit this enhancement.

DBRC improvements

The improvements in DBRC are:

- You can run DBRC under BPE.

  This improvement allows you to exploit BPE features for your existing DBRC user exits, such as refreshing them dynamically.

  It also brings the opportunity to collect DBRC performance data by writing a new (or modifying an existing) BPE statistics exit.
- If you have implemented RECON security, IMS 11 enables you to more easily use copies of the RECON that you might have made for recovery or problem diagnosis.
- The following new command helps you remove obsolete data from the RECONs and which would be tedious to remove by other means:

  `CLEANUP.RECON`

These improvements are discussed in Chapter 5, “DBRC enhancements” on page 147.

Miscellaneous

If you have ODBA programs that connect to more than one IMS at the same time, a change to the Call IMS (CIMS) might interest you. Refer to 3.8, “ODBA enhancements” on page 95.

Other useful improvements to database manageability are discussed in 3.4, “Database RAS” on page 90 and 3.7, “Fast Path usability” on page 94.

1.3.5 Transaction Manager manageability

This section discusses several changes to the IMS 11 Transaction Manager.

Type-2 command improvements

The type-2 command improvements include the ability to:

- Query transaction manager resources (namely LTERMs, nodes, users and user IDs); refer to 4.1, “Type-2 QUERY commands” on page 98.
- Control OTMA descriptors with CREATE, DELETE, QUERY and UPDATE commands. In particular, this enhancement enables you to create a new OTMA descriptor without having to stop and start IMS. For details, refer to 4.4, “OTMA type-2 commands” on page 127.
- Query OTMA transaction instances. This enhancement is also in 4.4, “OTMA type-2 commands” on page 127.

OTMA improvements

Improvement to OTMA include:

- DFS555I messages are returned to an OTMA client even if the transaction failed on the server system.
- If an IMS Connect client does not acknowledge a CM0 (commit then send) response, IMS 11 moves the unacknowledged response to a special hold-queue (the timeout queue), freeing the transaction pipe (tpipe) for further traffic.
- OTMA now provides state information about a periodic basis to its clients.

  IMS Connect has been enhanced to save this information at data store level, pass it to clients on request, and create events containing this information.

  IBM IMS Connect Extensions is able to process the data store status.

Refer to 4.3, “OTMA enhancements” on page 112.

Exit DFSMSCE0

Additional options in Multiple Systems Coupling (MSC) exit DFSMSCE0 include:

- You can route a transaction to a particular member of a shared queues group (this process is called transaction affinity).
- New ways are available to add accounting information to outbound messages.

Refer to 4.6, “Multiple Systems Coupling enhancements” on page 140 for more details.

Miscellaneous

IMS 11 can monitor the LUMC[1] and LUMP[2] pools (used with Advanced Program-to-Program Communication, or APPC) to determine that they are not running out of storage. IMS only does this if both pools are defined with more than 8 MB of storage and if they are not defined as 2 GB.

For more information, refer to 4.2, “APPC enhancements” on page 111.

A new buffer pool, DYNP, is used by OTMA and APPC for dynamic storage.

Refer to 4.9, “Dynamic storage private buffer pool enhancement” on page 146.

1 Value for the upper expansion limit of the LU 6.2 device manager common buffer pool
2 Value for the upper expansion limit of the LU 6.2 device manager private buffer pool

Full function response mode can be recovered across an IMS restart. Before version 11, IMS preserved the reply message, but returned it asynchronously and so it might not be the first message the user saw on reconnection. This item is retrofitted to IMS 9 and IMS 10 by a PTF. Refer to 4.7, “Full function response mode recovery” on page 144 for details.

1.4 Scalability

IMS 11 improves the already impressive scalability of IMS, chiefly by further exploiting storage above the 2 GB bar.

1.4.1 Fast Path 64-bit buffer manager

You may choose to store DEDB data buffers in 64-bit storage. If you do this, the new Fast Path 64-bit buffer manager will manage the buffer pool autonomically for you up to the size you originally specified.

Using the new UPDATE POOL command, you can dynamically increase the size of the 64-bit DEDB data buffer pool.

Furthermore, this new buffer pool manager allows you to have several subpools with different control interval (CI) sizes. This approach offers the possibility of retuning a DEDB to a more efficient CI size without affecting the whole system.

Refer to 3.6, “Fast Path 64-bit buffer manager” on page 91.

1.4.2 ACB caching

IMS 11 introduces the possibility to cache ACB library members in 64-bit storage. This enhancement does not replace the existing program specification block (PSB) and DMB pools, but instead it acts as a buffer for the ACBLIB, thereby reducing ACBLIB I/O for often-used member because these are found in the cache.

Refer 3.3.1, “Caching ACBs” on page 88.

1.4.3 Miscellaneous

HALDB Online Reorganization has several performance enhancements that enable you to run OLR with less impact to your system. Refer to 3.5, “OLR performance” on page 90.

IMS 11 changes the IMODULE internal service to optionally use 64-bit private storage instead of 24-bit common storage. This change improves scalability because common storage below the 16 MB line is an extremely constrained resource. Refer 2.5, “LSQA storage reduction” on page 40.

IMS 11 includes a change in the scheduling algorithm. This change is intended to reduce the number of false schedules in a shared queues environment where most transactions are local first. For more information, refer to 4.8, “Shared queues scheduling enhancement” on page 145.

1.5 Enhancements in IMS 10 if migrating from IMS 9

This section summarizes the main enhancements in IMS 10, which are new to you if you are migrating from IMS 9. Other enhancements that are not discussed in this book are described in *IBM IMS Version 10 Implementation Guide: A Technical Overview*, SG24-7526.

1.5.1 Integration

IMS 10 extended the integration of IMS, as discussed in this section.

Synchronous callout

IMS 10 introduced synchronous callout to Web services. Because this was introduced by a PTF after the release of IMS 10, this book discusses it in 6.7.1, “Synchronous call-out” on page 215.

Composite business application support

With versions prior to IMS 10, an IMS Connect client driving a conversational transaction had to use the same socket on all iterations. This approach was because IMS Connect saved the conversation token, associating it with the socket. With versions after IMS 10, the client can pass the conversation token to IMS Connect, freeing itself to use any socket.

1.5.2 Openness

IMS applications written in Java using the IMS DB Resource Adapter can exploit XQuery. IMS 11does this by adding a second parameter to the `retrieveXML` call.

1.5.3 Manageability

IMS 10 had many manageability improvements, especially in the Transaction Manager. These improvements are discussed in this section.

Dynamic resource definition

IMS 10 allowed you to define and modify programs, databases transactions and Fast Path routing code by using Type-2 commands. You no longer require the MODBLKS system generation process, MODBLKS data sets, or online change for MODBLKS.

Transaction level statistics

IMS 10 introduced the option to record statistics at an individual transaction level. Previous IMS versions record statistics at the scheduling level, which generally covers several instances of the transaction.

SMU removal

IMS 10 removed the IMS internal security. All security is now handled through System Authorization Facility (SAF); IMS no longer uses the MATRIX data sets.

ACB Member Online Change

IMS 10 allowed you replace selected ACB members in the active ACBLIB. While the change is happening, only those members are affected. Work continues to process for transactions that do not require the changed members. Member online change is part of global online change.

Operations Manager

IMS introduced the ability for Operations Manager (OM) to handle unsolicited messages and an optional OM audit log, which is a z/OS log stream. OM API programs can query the unsolicited messages and react accordingly.

Also new with IMS 10 was the ability to enter transactions or messages from a TSO single point of control (SPOC) using the QUEUE TRAN or QUEUE LTERM commands. The following command provided, for the first time, the ability to delete a transaction on a non-shared queue system:

```
QUEUE TRAN NAME(tran) OPTION(DEQ1 | DEQALL)
```

Sysplex Resource Management

IMS 10 allowed you to store the status of transactions, areas, and databases in the RM CF structure so that an IMS that is restarting can know about changes that happened while it was down (typically for a planned outage).

If you are using the RM structure, IMS 10 automatically enforces serially running of programs that are defined as SCHDTYP=SERIAL.

DBRC API

IMS 10 extended the DBRC API to allow the BACKUP, INIT, CHANGE, DELETE, GENJCL NOTIFY, and RESET commands. Although IMS 9 already provided the equivalent of LIST commands, IMS 10 extended it.

You can also register your program as a subsystem, which can be useful if you have programs that manipulate databases in ways that IMS would not otherwise know about.

Read-only DBRC access

IMS 10 allowed you to access the RECONs in read-only mode. Earlier versions of IMS required control access even for query activity.

Change accumulation process

The change accumulation (CA) process was modified so that CA JCL, which you generated, was not necessarily invalidated by later changes to the RECON.

There was also support for the concept of a *point-in-time* CA. This concept is not implemented by IMS, but is implemented in IBM IMS High Performance Change Accumulation and IBM IMS Database Recovery Facility.

Destination routing descriptors

IMS 10 introduced a new descriptor type in PROCLIB member DFSYDTx. It allowed you to:

- Associate a destination with a particular tpipe of a particular IMS Connect.
- Identify, as non-OTMA, a destination that would otherwise be treated as OTMA (such as alternate PCB output from an OTMA-sourced transaction).

Previous IMS versions required code in exits DFSYPRX0, DFSYDRU0, or both, to identify those associations.

IRLM time out

IMS 10 introduced the possibility for an application to get a DLI status code instead of an abend when IRLM times it out.

Abend search and notification

Optionally, IMS 10 can send you an e-mail when it abends. This e-mail contains diagnostics and links to IBM support information.

1.5.4 Scalability

This section lists the scalability enhancements in IMS 10.

Parallel RECON Access

Busy data-sharing IMS systems can be slowed when DBRC performs RECON access serially. Parallel RECON Access (PRA) is an option that removes this restriction. PRA requires Transactional VSAM, which is a chargeable feature of DFSMS.

DBRC time stamp precision

Earlier versions of IMS use a time stamp that is accurate to 0.1 seconds. On a fast system, this feature could (in the near future) lead to duplicate time stamps. IMS 10 extended the time stamps to microsecond precision to avoid this problem.

ILDS Rebuild performance

IMS 10 added a new option to the a HALDB indirect list data set (ILDS) Rebuild utility DFSPREC0 that can significantly improve its performance. It uses VSAM load mode (instead of update mode) and so it also leaves usable free space in the ILDS.

DEDB buffers

IMS 10 increased the maximum number of buffers from 65535 to 2^{32}-1 because the previous limit was becoming a problem for busy Fast Path systems.

MSC bandwidth mode

IMS 10 improved the blocking of MSC messages to increase throughput.

Large data sets

IMS 10 was the first version to support data sets over 65535 tracks (`DSNTYPE=LARGE` introduced by z/OS 1.7). It covered OSAM databases, OLDS, message queue data sets and GSAM files.

2

System enhancements

This chapter describes the major enhancements for the overall IMS 11 system, including descriptions of the main components:

- User Exit Interface
- IMS Interactive Dump Formatter
- Installation verification procedure (IVP)
- Syntax Checker
- LSQA storage reduction
- KBLA
- The /DIAGNOSE command
- Dynamic Abend Dump Formatting exit

2.1 User Exit Interface

The user exit enhancements introduce three exit types (with sample exit routines), and enhancements for the `/TRA` command for DFSMSCE0.

Also, when exit routines which were available before IMS 11 have to be modified or brought offline, IMS has to be stopped and restarted to recognize the changed or new exit routines. This restriction does not apply to the three new exit routine types introduced in IMS 11.

The user exit enhancements solve this problem with two type-2 commands:

- REFRESH USEREXIT
- QUERY USEREXIT

These commands can be invoked by any method that utilizes the OM API (for example, the TSO SPOC, the IMS Control Center, and automation programs).

These enhancements are delivered through the IMS Version 11 service process.

2.1.1 Overview

You can refresh certain exit types without stopping IMS. The exit routines are loaded and are available for use while IMS continues to process work.

The enhancement applies to the following exit routines types introduced in IMS 11:

- Early Initialization exit
- IMS CQS Event exit
- IMS CQS Structure Event exit

This enhancement also enables IMS to support multiple instances of a user exit type without requiring code that is written by the customer or vendor.

You can run multiple tools that use the same exit point without the need for additional software to manage the multiple exits. This enhancement provides this exit management for exit types that are supported.

The enhancements are planned to be delivered through the IMS 11 service process and subject to change.

Product Partner exit routine (DFSPPUE0)

Additional support for the new user exit services is added for the Product Partner exit routine (DFSPPUE0). This exit routine is entered immediately before IMS is ready for startup (before the DFS994I start-complete message is issued).

The exit routine allows initialization of products that run with IMS. It can load or link one or more partner product routines.

Standard user exit parameter list

Enhancements to the standard user exit parameter list (SXPL) include adding a pointer to a static work area.

Many of the IMS user exit routines use a standard user exit interface. This interface allows the exit routines to use callable services to access IMS blocks. At the same time, this interface creates a clearly differentiated programming interface (CDPI) between IMS and the exit routine. Part of the interface consists of a standard user exit parameter list.

A new sub code that contains relevant data about the new exit routines is added to the X'45' statistics log record.

The new type-2 commands have the same dependencies as other type-2 commands. This includes a minimum CSL (SCI and OM) and an application such as the TSO SPOC from which commands can be issued.

2.1.2 REFRESH USEREXIT command

The REFRESH USEREXIT type-2 command refreshes the user exit types that are defined in the USER_EXITS section of the DFSDFxxx member. When the command is processed, IMS rereads the DFSDFxxx member and processes the USER_EXITS section.

Use the REFRESH USEREXIT type-2 command to bring new or modified exit routines online (and delete the old exit routines) without requiring that IMS be stopped and restarted.

After the new or modified exit routines are online, any subsequent calls to those exit types result in the execution of the new or modified exit routines.

Only exit types that are specified on the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member of the IMS PROCLIB data set are eligible for refreshing.

When the REFRESH USEREXIT is entered, IMS performs the following steps:

1. Reads the DFSDFxxx member and processes the USER_EXITS section.
2. Loads the user exit modules specified in the USER_EXITS section for the exit types specified in the command.
3. Updates the internal IMS control block with pointers to the new user exit modules. Any subsequent calls to the user exit modules will now call the new modules.
4. When the processing has completed in the old exit modules, deletes the old modules.

Process

IMS loads the new user exit modules before deleting the old modules. If an error occurs during this process (for example, a module could not be loaded), IMS fails the command for the particular user exit type and leaves the current modules of the user exit type in effect.

All modules of the specified user exit type must be loaded successfully for the command to complete successfully.

The REFRESH USEREXIT command is routed to each IMS in the IMSplex as the default routing. This process sends the REFRESH command to each IMS in the IMSplex and the user exit modules are refreshed in each IMS.

If the refresh fails on one or more IMSs, you must resolve the problem that caused the command to fail, and then reenter the command.

The REFRESH USEREXIT command is valid for all control region types.

Parameter

As mentioned, the REFRESH USEREXIT type-2 command refreshes the modules for the specified user exits. This means that the command can only be specified through the Operations Manager (OM) and that the output of the command is delivered in XML and is available to automation programs that communicate with OM.

As mentioned previously, the REFRESH USEREXIT command is routed to each IMSs in the IMSplex by default.

The command has one parameter, TYPE(), which specifies the user exit type or types that you want to be refreshed. You can specify a single user exit type or a list of user exit types separated by commas.

The valid user exit types are in "DFSDFxxx changes" on page 23.

Output fields

Table 2-1 lists the REFRESH USEREXIT output fields. The columns in the table are:

- Short label: Contains the short label generated in the XML output.
- Keyword: Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.
- Scope: Identifies the scope of the output field.
- Meaning: Provides a brief description of the output field.

Table 2-1 REFRESH USEREXIT output fields

Short label	Keyword	Scope	Meaning
CC	N/A[a]	N/A	Completion code for the line output
MBR	N/A	N/A	IMSPLEX member that built the output line
TYPE	TYPE	LCL[b]	User exit type that is specified by the REFRESH command
NAME	N/A	LCL	Exit routine that was loaded by this REFRESH command

a. N/A (not applicable) appears for output fields that are always returned.
b. LCL ((local) indicates that the field can be generated by any IMS displaying information for SHOW(LOCAL).

Table 2-2 lists the return and reason codes, for the REFRESH USEREXIT command.

Table 2-2 REFRESH USEREXIT return and reason codes

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001010'	The command is not processed because no user exit types were found that matched the TYPE parameter.
X'0000000C'	X'00003000'	The command was not successful for one or more user exit types.
X'0000000C'	X'00003004'	The command was not successful for all the user exit types specified.
X'00000014'	X'00005004'	The command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The command processing terminated because of an internal error.

Table 2-3 lists completion codes that can be returned on a REFRESH USEREXIT command, and it indicates the reason for the error with the user exit type.

Table 2-3 REFRESH USEREXIT completion codes

Completion code	Meaning
0	The command completed successfully for the user exit type.
10	The user exit type is unknown to the IMS that is processing the command. Confirm that the user exit type is correctly spelled.

RACF definitions for command protection

Table 2-4 shows the RACF definitions in class OPERCMDS necessary to protect REFRESH USEREXIT.

Table 2-4 RACF definitions to protect REFRESH USEREXIT

Command	Keyword	Access required	Resource name
REFRESH	*USEREXIT*	*UPDATE*	*IMS.plexname.REFRESH.USEREXIT*

Example

Example 2-1 shows sample TSO SPOC input and output of the REFRESH USEREXIT command.

Example 2-1 REFRESH USEREXIT command

```
TSO SPOC input:
REFRESH USEREXIT TYPE(INITTERM)

TSO SPOC output:
  Response for: REFRESH USEREXIT TYPE(INITTERM)
TYPE            MBRNAME       CC     NAME
INITTERM        IMS1          0      MYINTRM2
INITTERM        IMS1          0      MYINTRM1
INITTERM        IMS2          0      MYINTRM2
INITTERM        IMS2          0      MYINTRM1
```

2.1.3 QUERY USEREXIT command

The QUERY USEREXIT type-2 command displays information about the exit routines that are defined in the USER_EXITS section of the DFSDFxxx member of the IMS PROCLIB data set.

Information about exit routines that are not specified in the USER_EXITS section of the DFSDFxxx member is not displayed in the output of the QUERY USEREXIT command.

The output contains an entry for each user exit module within each user exit type named in the QUERY.

The QUERY USEREXIT command is valid for all control region types.

Parameters

The QUERY USEREXIT type-2 command can only be specified through the Operations Manager (OM). The command output is delivered in XML and is available to automation programs that communicate with the OM.

The QUERY USEREXIT command is routed each IMS in the IMSplex by default and it has the following parameters:

- TYPE()

  Specifies the user exit type or types for which you want information displayed. You can specify a single user exit type or a list of user exit types separated by commas.

  The valid user exit types are in “DFSDFxxx changes” on page 23.

- SHOW()

  Specifies information about the user exit routines to be returned in the output fields of the command response. The exit type and module name fields are always returned with the name of the IMS that created the output for the user exit type and the completion code.

  The valid fields that can be specified are:

  - ALL

    All possible output fields are returned.

  - ACTIVE

    The number of currently active instances of the user exit routine is a point-in-time value that represents the number of calls to the user exit that are still in progress and have not returned to IMS.

  - CALLS

    This is the number of calls to the user exit since the last user exit routine refreshed.

    For performance reasons, serialization is not obtained when IMS collects this number. For an exit type that can run multiple instances in parallel, this number is an approximation.

    The maximum value that can be displayed in this field is 2147483647 (2^{31}-1). If the call count exceeds this value, 2147483647 is displayed.

  - ENTRYPT

    This option is the entry point address of the user exit routine.

  - ETIME

    This option is total (cumulative) elapsed time in milliseconds spent in the exit module since it was last refreshed.

    For performance reasons, serialization is not obtained when IMS collects this number. For an exit type that can run multiple instances in parallel, this number is an approximation.

    The maximum value that can be displayed in this field is 2147483647 (2^{31}-1). If the call count exceeds this value, 2147483647 is displayed.

  - LOADPT

    This option is the address at which the user exit routine was loaded.

  - RTIME

    This option is the local date and time that the user exit routine was last refreshed (or initially loaded, if no refreshes have occurred). The format of the output field is:

    ```
    yyyy-mm-dd hh:mm:ss.th
    ```

– SIZE

  This option is the size in bytes of the user exit load routine. This value is displayed in hexadecimal.

– TEXT

  This option is 32 bytes starting from offset +04 from the exit module's entry point, translated to EBCDIC with non-printable characters replaced by the period (.) character.

  This location is common for module identification information. If your user exit routines contain printable identification data at this point in the module, the TEXT option enables that information to be displayed.

Output fields

Table 2-5 lists the QUERY USEREXIT output fields. The columns in the table are:

- Short label: Contains the short label generated in the XML output.
- Keyword: Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.
- Scope: Identifies the scope of the output field.
- Meaning: Provides a brief description of the output field.

Table 2-5 QUERY USEREXIT output fields

Sort label	Keyword	Scope	Meaning
CC	N/A	N/A	Completion code for the line of output
MBR	N/A	N/A	IMSPLEX member name that built the output line
TYPE	TYPE	LCL	User exit type requested by the QUERY command
NAME	N/A	LCL	User exit module name
ACTIVE	ACTIVE	LCL	Number of active instances of this exit module
CALLS	CALLS	LCL	Number of calls to this user exit module since last refresh
ENTRYPT	ENTRYPT	LCL	The entry point of this user exit module
ETIME	ETIME	LCL	Total time spent in this user exit module since last refresh
LOADPT	LOADPT	LCL	The load point of this user exit module
RTIME	RTIME	LCL	The time this user exit module was last refreshed
SIZE	SIZE	LCL	The size in hexadecimal of this user exit module
TEXT	TEXT	LCL	32 bytes from this user exit module translated into EBCDIC

Return, reason, and completion codes

Table 2-6 on page 20 lists the return and reason codes for the QUERY USEREXIT command.

Table 2-6 QUERY USEREXIT return and reason Codes

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001010'	The command is not processed because no user exit routines were found that matched the TYPE parameter.
X'0000000C'	X'00003000'	The command was not successful for one or more user exit types.
X'0000000C'	X'00003004'	The command was not successful for all the user exit types specified.
X'00000014'	X'00005004'	The command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The command processing terminated because of an internal error.

Table 2-7 lists completion codes that can be returned on a QUERY USEREXIT command, and it indicates the reason for the error with the user exit type.

Table 2-7 QUERY USEREXIT completion codes

Completion code	Meaning
0	The command completed successfully for the exit routine.
10	No resources were found. The resource is the user exit type. None of the user exit types specified are known to the IMS that processed the command. Confirm that the correct spelling of the user exit types is specified on the command.

RACF definitions for command protection

Table 2-8 shows RACF definitions in class OPERCMDS needed to protect QUERY USEREXIT.

Table 2-8 RACF definitions to protect QUERY USEREXIT

IMS command	Command keyword	Access required	Resource name
QRY	USEREXIT	READ	IMS.plexname.QRY.USEREXIT

QUERY USEREXIT example

Example 2-2 shows TSO SPOC input and output of the QUERY USEREXIT command.

Example 2-2 QUERY USEREXIT command

```
TSO SPOC input:
QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)

TSO SPOC output:
  Response for: QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)
TYPE            NAME         MBRNAME        CC   CALLS  RTIME
INITTERM        MYINTRM1     IMS1           0    1      2007-02-10 05:19:42.33
INITTERM        MYINTRM2     IMS1           0    1      2007-02-10 05:19:42.33
INITTERM        MYINTRM1     IMS2           0    1      2007-02-10 05:19:42.38
INITTERM        MYINTRM2     IMS2           0    1      2007-02-10 05:19:42.39
```

2.1.4 Initialization/Termination user exit

The Initialization/Termination exit is called during early IMS initialization, during normal IMS termination, and after the successful refresh of a user exit type.

The exit is available to all IMS regions including DB/DC, DBCTL, DCCTL, and FDBR.

This exit is called by IMS only if it is defined using the EXITDEF parameter with TYPE=INITTERM in the USER_EXITS section of the DFSDFxxx member. There is no default exit name. The exit routine or routines must be defined by the user.

This exit is also called after a successful REFRESH USEREXIT command with a function code 2: Redrive for exit refresh.

This approach allows the users to reformat storage that was allocated or to do any other tasks that must be done to allow new versions of other exit routines to be called.

If the INITTERM exit is successfully refreshed, the new copy of the exit routine (or routines) is called.

A sample of the Early Initialization exit routine is included with IMS 11.

2.1.5 IMS CQS Event exit type

The IMS CQS Event exit is called when IMS processes a CQS event. IMS is notified of the events from CQS through an IMS exit routine that is driven in the IMS address space by CQS.

This exit type is available to an IMS control region that registers with CQS, which can be in either a DB/DC or DCCTL configuration.

This exit type is called by IMS only if it is defined using the EXITDEF parameter with TYPE=ICQSEVNT in the USER_EXITS section of the DFSDFxxx member. There is no default exit name. The exit routine (or routines) must be defined by the user.

A sample CQS Event exit routine is included with IMS 11.

2.1.6 IMS CQS Structure Event exit type

The IMS CQS Structure Event exit type routine is called when IMS processes a CQS structure event. IMS is notified of the structure events from a CQS-driven Structure Event exit routine that is driven in the IMS address space by CQS.

This exit type is available to an IMS control region that registers with CQS, which can be in either a DB/DC or DCCTL configuration.

This exit routine is called by IMS only if it is defined using the EXITDEF parameter with TYPE=ICQSSTEV in the USER_EXITS section of the DFSDFxxx member. There is no default exit name. The exit routine (or routines) must be defined by the user.

A sample IMS CQS Structure Event exit routine is include with IMS 11.

2.1.7 Product Partner exit (DFSPPUE0)

Support for the new user exit services is added for the Product Partner exit (DFSPPUE0). You have the choice to continue to use the old user exit services or to use the new user exit services, which give the capability to define multiple user exit routines that are called at the PPUE exit point. The exit can be named DFSPPUE0 and linked into a library that is in the STEPLIB concatenation. The exit will be called by IMS.

Multiple exit routines can be defined to IMS using the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member. The routines are called in the order that they are listed in the EXITDEF parameter. If an exit is defined multiple times, it is called multiple times.

2.1.8 Standard user exit parameter list enhancements

The user exit enhancements in IMS 11 introduce version 6 of the standard user exit parameter list (SXPLVER6). The enhancement adds the following fields:

- A pointer to a 256-byte static work area (described in "Static work area" on page 22).
- The version of IMS that is calling the exit routine
- The four-character IMSID
- The eight-character Recoverability Service Name (RSENAME)

  This name is set by using the RSENAME startup parameter in the DFSHSBxx in PROCLIB. If the control region is not XRF or DBCTL warm-standby capable, this field is blank.
- A pointer to a byte in storage that the user exit uses to indicate whether to call the next user exit routine in the list.

  If the exit sets this byte, it must be one of the following values, which are defined in the DFSSXPL macro:

SXPL_CALLNXTY	Call the next exit.
SXPL_CALLNXTN	Do not call the next exit.

  If the exit routine does not set this byte, the default is to call the next user exit routine in the list.

Static work area

Each exit routine that uses the SXPL is assigned its own static work area. The work area is set to zeros before the exit routine is called for the first time.

After that, the work area is preserved, even if an exit routine is updated by using a REFRESH USEREXIT command. If an exit routine is deleted, the work area is deleted.

Exit routines that are brought online by using the REFRESH USEREXIT command are assigned new, zeroed work areas.

Because of the way the REFRESH USEREXIT works, if the exit routine is replaced with the command, the static work area might be accessed with both the old and new versions of the exit routine at the same time. You should consider this information when you design the access to your data area.

If the mapping of the static area has not changed, this is no different than concurrent executions of the exit routine accessing the static area.

If the mapping of the static area has changed, put a version in the mapping of the static area and ensure the exit routines are sensitive to the version.

The area is available for all user exits that use the SXPL whether or not they are defined using the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member.

2.1.9 Installing the user exit enhancements

The USER_EXITS section of the DFSDFxxx member was introduced in IMS 10 to define the modules that are called at the Restart exit point during IMS restart processing. This enhancement adds new user exit types.

DFSDFxxx changes

A user exit must be defined in the USER_EXITS section of the DFSDFxxx member to have information returned with the QUERY USEREXIT command and to have the ability to be refreshed with the REFRESH USEREXIT command.

The new user exit types allowed in the USER_EXITS section of the DFSDFxxx are:

INITTERM	Initialization/Termination exit type
ICQSEVNT	IMS CQS Event exit
ICQSSTEV	IMS CQS Structure Event exit
PPUE	IMS Product Partner User exit

2.1.10 DFSMSCE0 user exit trace enhancement

This section is about the trace command enhancement for DFSMSCE0. For a discussion of the enhancements to DFSMSCE0 itself, refer to 4.6, “Multiple Systems Coupling enhancements” on page 140.

/TRACE EXIT command

The DFSMSCE0 exit trace was enhanced to allow the GU entry point to be traced and displayed. The commands you can use are:

```
/TRACE SET ON|OFF EXIT PRGU DFSMSCE0
/DISPLAY TRACE EXIT
```

Figure 2-1 shows the syntax of the /TRA command for DFSMSCE0.

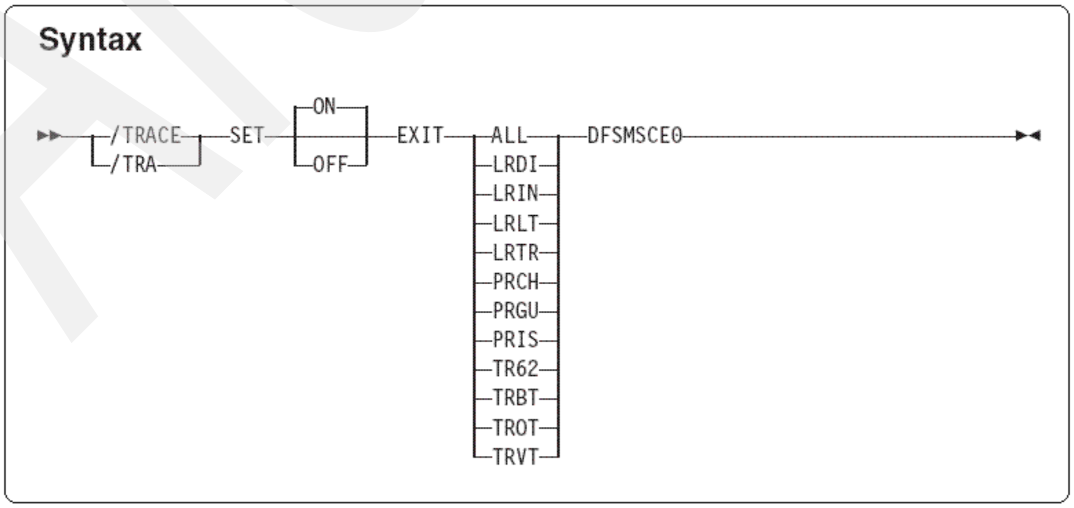


Figure 2-1 TRACE EXIT command syntax

This enhancement adds the keyword PRGU for calls to the Program Routing GU Call entry point. When the trace is set for an entry point, IMS writes the following log records:

- A 6701-MSEA record when the entry point is called.
- A 6701-MSEB record when the exit returns to IMS.

/DISPLAY TRACE EXIT command

Example 2-3 shows a sample /DISPLAY TRACE EXIT command. ON or OFF means the exit is being used and the trace is on or off for the entry point.

Example 2-3 /DISPLAY TRACE EXIT command showing ON and OFF values

```
/DIS TRACE EXIT
DFS4444I DISPLAY FROM ID=IM1B
    IMS ACTIVE TRACES

       EXIT       FUNC   STATUS
     DFSMSCE0     TRBT     ON
     DFSMSCE0     TRVT     ON
     DFSMSCE0     TR62     OFF
     DFSMSCE0     TROT     OFF
     DFSMSCE0     LRTR     ON
     DFSMSCE0     LRLT     ON
     DFSMSCE0     LRDI     ON
     DFSMSCE0     LRIN     OFF
     DFSMSCE0     PRCH     OFF
     DFSMSCE0     PRIS     ON
     DFSMSCE0     PRGU     ON
    *09239/233728*
```

Example 2-4 shows another sample of the /DISPLAY TRACE EXIT command. N/A means the entry point is not being used.

Example 2-4 /DISPLAY TRACE EXIT command showing N/A value

```
/DIS TRACE EXIT
DFS4444I DISPLAY FROM ID=IM1B
     IMS ACTIVE TRACES

        EXIT       FUNC   STATUS
      DFSMSCE0     TRBT     N/A
      DFSMSCE0     TRVT     N/A
      DFSMSCE0     TR62     N/A
      DFSMSCE0     TROT     N/A
      DFSMSCE0     LRTR     N/A
      DFSMSCE0     LRLT     N/A
      DFSMSCE0     LRDI     N/A
      DFSMSCE0     LRIN     N/A
      DFSMSCE0     PRCH     N/A
      DFSMSCE0     PRIS     N/A
      DFSMSCE0     PRGU     N/A
     *2009239/182044*
```

Note: Status ON, OFF, N/A can be mixed.

2.2 IMS Interactive Dump Formatter

IMS Interactive Dump Formatter is enhanced in IMS 11 to re-create the final part of an IMS log from the information that is available in an IMS dump, thus eliminating having to request the final system log data set (SLDS) for diagnostic purposes.

2.2.1 Overview

When an IMS customer is working with an IBM Software Support representative, frequently the customer creates a memory dump and the service representative needs IMS log data.

To get this log data, the customer requests the final SLDS, compresses (terses) it, and transfers the file to IBM by using FTP. The Service person decompresses (unterses) the data set and then analyzes it.

The enhancements built into the IMS Interactive Dump Formatter in IMS 11 can build a log data set from the log records that reside in the dump's log buffers, thus avoiding the requests for the SLDS and the extra work associated with these requests.

2.2.2 Description

The purpose of this enhancement is to recreate the final part of an IMS log from the information that is available in an IMS dump.

The log records that reside in the dump's log buffers are copied into a variable block file (RECFM=VB) that can be analyzed by KBLA, DFSERA10 or other log analysis tools.

From there, log records can be put in increasing log sequence number (LSN) order and be manipulated by existing KBLA functions or DFSERA10. This function is provided from a new panel in the Enhanced Dump Analysis (EDA) option of the Dump Formatter so that the log records still residing in the dump's log buffers can be written to an output data set.

When choosing the WRITE option from the SYSTEM panel of the EDA option, the user will input two data set names. The first is the name of the data set that will contain the log records written from the dump. The second is the name of a data set that will contain report data, such as the address of each log buffer and the sequence numbers of the log records.

Also included at the end of the report data set is the number of log records that were written to the data set. If either of these data sets has not been created, then one is created for the user with the name specified.

The log records can be extracted from dump buffers by using the IPCS dialog or IPCS in batch.

2.2.3 Extracting log records using the IPCS

Log records can be written to an output data set by navigating through the IPCS panels of the IMS Dump Formatter. To extract the log records from the dump by using the Interactive Problem Control System (IPCS), follow these steps:

1. Select option **0** from the IPCS Primary Option menu to specify the dump data set to analyze.
2. Specify the data set name for the dump data set. Figure 2-2 on page 26 shows this selection.

```
-------------------------IPCS Default Values ---------------------------
Command ====>

 You may change any of the defaults listed below.  The defaults shown before
any changes are LOCAL.  Change scope to GLOBAL to display global defaults.

Scope   ==> LOCAL   (LOCAL, GLOBAL, or BOTH)

If you change the Source default, IPCS will display the current default
Address Space for the new source and will ignore any data entered in
the Address Space field.

Source  ==> DSNAME(‘dump_data_set_name’)
Address Space   ==> ASID(X'029B')
Message Routing ==> NOPRINT TERMINAL
Message Control ==> NOCONFIRM VERIFY FLAG(TERMINATING)
Display Content ==> NOMACHINE REMARK REQUEST STORAGE NOSYMBOL
```

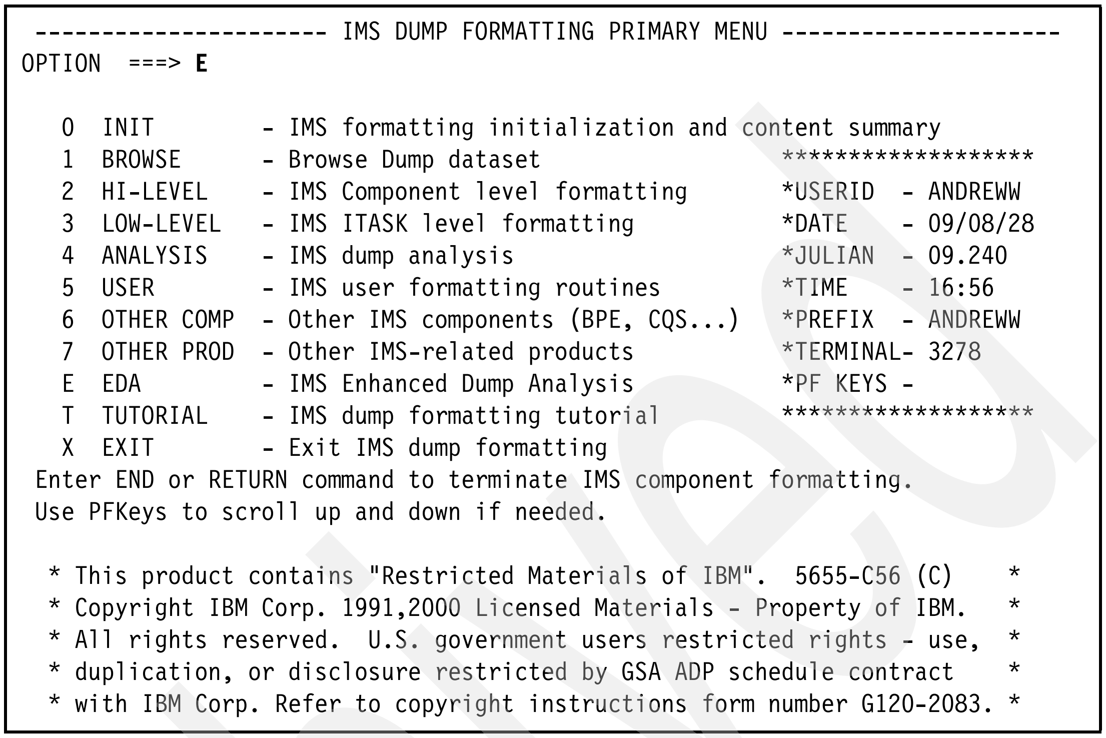
Figure 2-2 IPCS Dump data set name

3. Select option **2.6** from the IPCS Primary Option menu to display a list of the dump component analysis tools.
4. Select option **DFSAAMPR** from the IPCS MVS Dump Component Data Analysis menu to display options for the IMS Interactive Dump Formatter. Figure 2-3 shows this selection.

```
 ------------------ IPCS MVS DUMP COMPONENT DATA ANALYSIS -------------

To display information, specify "S option name" or enter S to the left
of the option desired.  Enter ? to the left of an option to display
help regarding the component support.

  Name     Abstract
  ALCWAIT  Allocation wait summary
  AOMDATA  AOM analysis
  APPCDATA APPC/MVS Data Analysis
  ASCHDATA APPC/MVS Scheduler Data Analysis
  ASMCHECK Auxiliary storage paging activity
  ASMDATA  ASM control block analysis
  AVMDATA  AVM control block analysis
  CICS330  CICS Version 3 Release 3 analysis
  COMCHECK Operator communications data
  COUPLE   XCF Coupling analysis
  CTRACE   Component trace summary
  DAEDATA  DAE header data
  DB2DATA  DB2 analysis
S DFSAAMPR IMS Interactive Dump Formatter
  DIVDATA  Data in virtual storage
  DLFDATA  Data Lookaside Facility data
```

Figure 2-3 IPCS MVS Dump Component Data Analysis options

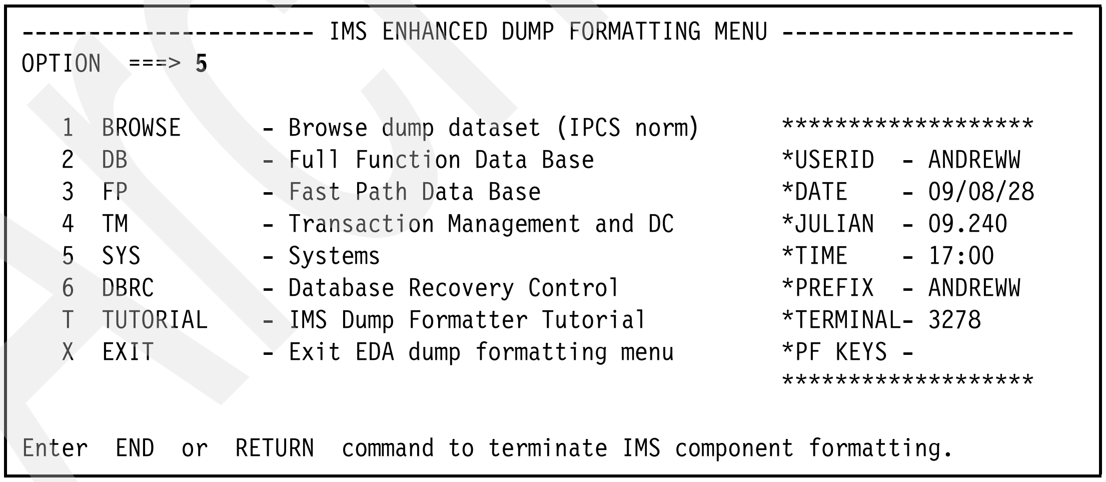
5. Select option **E** from the IMS Dump Formatting Primary menu to display options for IMS Enhanced Dump Analysis. Figure 2-4 shows this selection.



```
 ---------------------- IMS DUMP FORMATTING PRIMARY MENU ---------------------
OPTION  ===> E

   0  INIT        - IMS formatting initialization and content summary
   1  BROWSE      - Browse Dump dataset                    *******************
   2  HI-LEVEL    - IMS Component level formatting        *USERID  - ANDREWW
   3  LOW-LEVEL   - IMS ITASK level formatting            *DATE    - 09/08/28
   4  ANALYSIS    - IMS dump analysis                     *JULIAN  - 09.240
   5  USER        - IMS user formatting routines          *TIME    - 16:56
   6  OTHER COMP  - Other IMS components (BPE, CQS...)    *PREFIX  - ANDREWW
   7  OTHER PROD  - Other IMS-related products            *TERMINAL- 3278
   E  EDA         - IMS Enhanced Dump Analysis            *PF KEYS -
   T  TUTORIAL    - IMS dump formatting tutorial          *******************
   X  EXIT        - Exit IMS dump formatting
 Enter END or RETURN command to terminate IMS component formatting.
 Use PFKeys to scroll up and down if needed.

  * This product contains "Restricted Materials of IBM".  5655-C56 (C)    *
  * Copyright IBM Corp. 1991,2000 Licensed Materials - Property of IBM.   *
  * All rights reserved.  U.S. government users restricted rights - use,  *
  * duplication, or disclosure restricted by GSA ADP schedule contract    *
  * with IBM Corp. Refer to copyright instructions form number G120-2083. *
```

Figure 2-4 IMS Dump Formatting Primary Menu

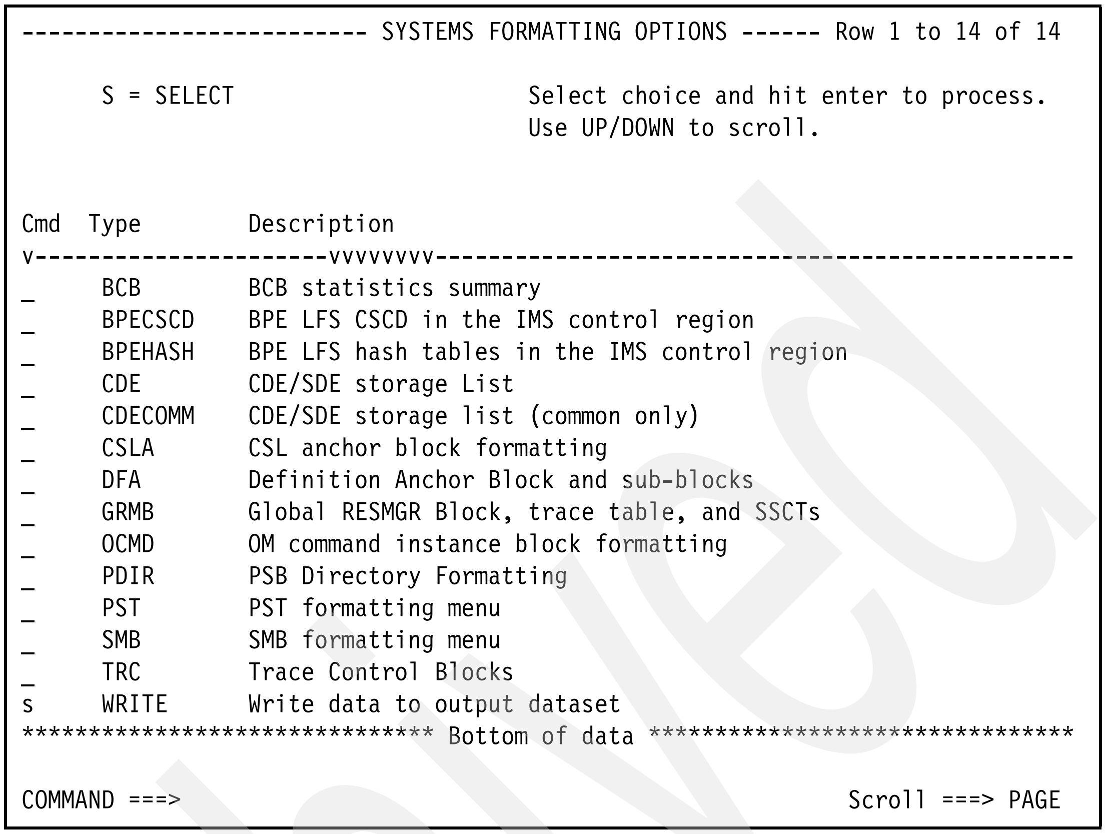
6. Select option **5** from the IMS Enhanced Dump Formatting Menu, for the Systems option. Figure 2-5 shows this selection.



```
---------------------- IMS ENHANCED DUMP FORMATTING MENU ----------------------
OPTION  ===> 5

   1  BROWSE      - Browse dump dataset (IPCS norm)       *******************
   2  DB          - Full Function Data Base               *USERID  - ANDREWW
   3  FP          - Fast Path Data Base                   *DATE    - 09/08/28
   4  TM          - Transaction Management and DC         *JULIAN  - 09.240
   5  SYS         - Systems                               *TIME    - 17:00
   6  DBRC        - Database Recovery Control             *PREFIX  - ANDREWW
   T  TUTORIAL    - IMS Dump Formatter Tutorial           *TERMINAL- 3278
   X  EXIT        - Exit EDA dump formatting menu         *PF KEYS -
                                                          *******************


Enter  END  or  RETURN  command to terminate IMS component formatting.
```

Figure 2-5 IMS Enhanced Dump Formatting options

7. Select the **WRITE** option from the Systems Formatting menu, to write data to an output data set and display the IMS EDA Get Output Data Set Panel. Figure 2-6 on page 28 shows this selection.

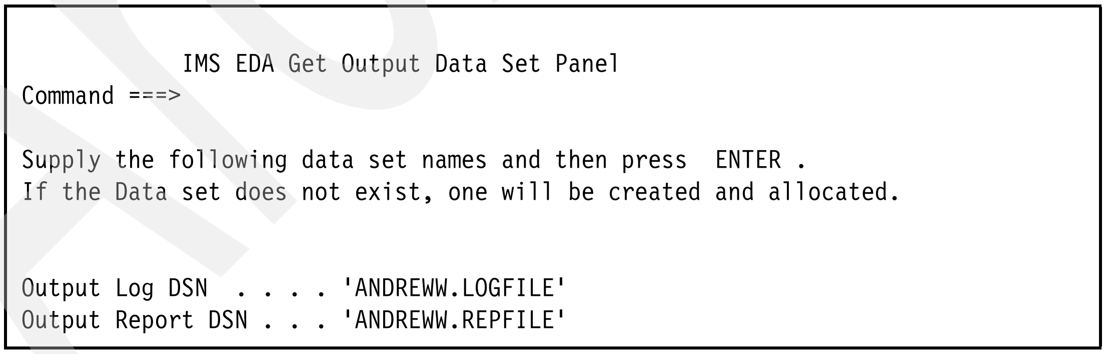


```
------------------------- SYSTEMS FORMATTING OPTIONS ------ Row 1 to 14 of 14

     S = SELECT                      Select choice and hit enter to process.
                                     Use UP/DOWN to scroll.


Cmd  Type       Description
v----------------------vvvvvvvv-----------------------------------------------
_     BCB        BCB statistics summary
_     BPECSCD    BPE LFS CSCD in the IMS control region
_     BPEHASH    BPE LFS hash tables in the IMS control region
_     CDE        CDE/SDE storage List
_     CDECOMM    CDE/SDE storage list (common only)
_     CSLA       CSL anchor block formatting
_     DFA        Definition Anchor Block and sub-blocks
_     GRMB       Global RESMGR Block, trace table, and SSCTs
_     OCMD       OM command instance block formatting
_     PDIR       PSB Directory Formatting
_     PST        PST formatting menu
_     SMB        SMB formatting menu
_     TRC        Trace Control Blocks
s     WRITE      Write data to output dataset
******************************* Bottom of data ********************************

COMMAND ===>                                                  Scroll ===> PAGE
```

Figure 2-6 Systems Formatting Option panel

8. When the The IMS EDA Get Output Data set panel opens, specify the data set to write the log records to and a data set for the report information.

   The report data set contains log sequence numbers for the log records written as well as the total number of log records written to the output data set. Figure 2-7 shows the IMS EDA Get Output Data Set Panel.



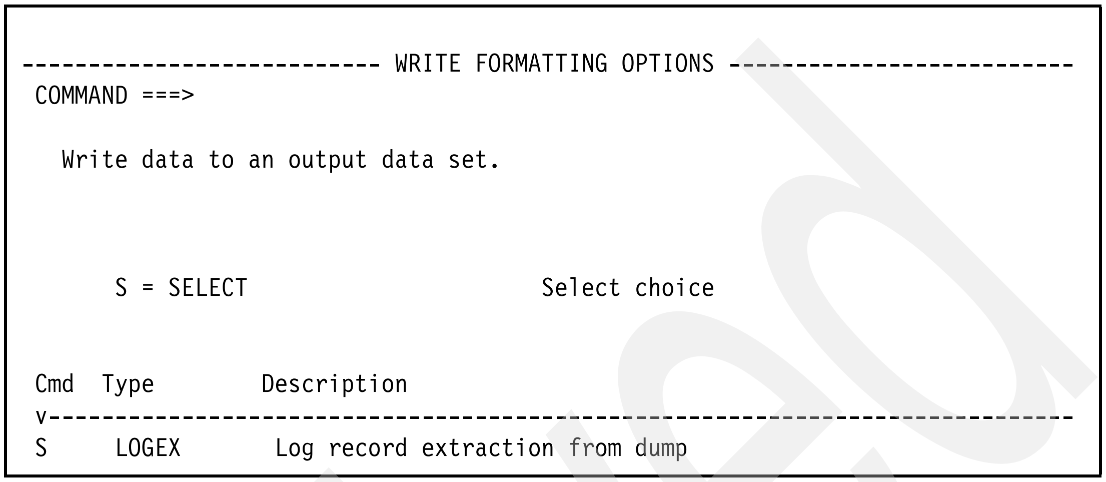
```
            IMS EDA Get Output Data Set Panel
Command ===>

Supply the following data set names and then press  ENTER .
If the Data set does not exist, one will be created and allocated.


Output Log DSN  . . . . 'ANDREWW.LOGFILE'
Output Report DSN . . . 'ANDREWW.REPFILE'
```

Figure 2-7 IMS EDA Get Output Data Set panel

If the specified data set names do not exit, they will be allocated with the following attributes of sequential data set:

– Output log data set, LRECL(32756) BLKSIZE(32760) RECFM(VB)
– Output Report data set, LRECL(133) BLKSIZE(1330) RECFM(FBA)

9. Select the LOGEX option from the Write Formatting Options panel to write the log records from the dump's buffers to the output data set that you specified in the previous step. Currently, the only choice is `Log record extraction from dump`. Figure 2-8 shows the selection in this panel.



```
--------------------------- WRITE FORMATTING OPTIONS ---------------------------
 COMMAND ===>

   Write data to an output data set.



       S = SELECT                      Select choice


 Cmd  Type        Description
 v------------------------------------------------------------------------------
 S     LOGEX       Log record extraction from dump
```

Figure 2-8 Write Formatting Options panel

When you return to the Systems Formatting Menu, you will get confirmation that the log records were written to the data set you specified along with any report information to the report data set specified.

2.2.4 Extracting log records using IPCS in batch

Example 2-5 on page 30 shows JCL for extracting the log records from the dump using a batch job.

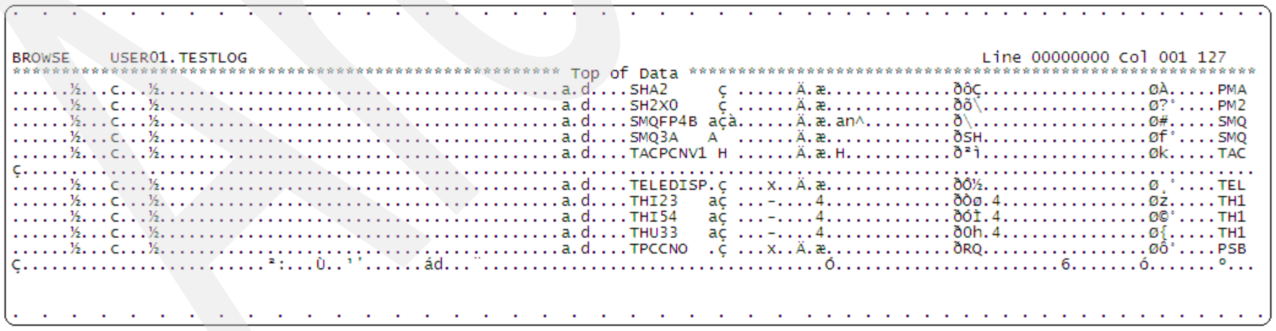
Before you begin extracting the log records from the dump, be sure you can access the dump data set from which you want to use to extract the log records. You might have to copy the dump data set to one that you can access from Interactive Problem Control System (IPCS).

Example 2-5 Batch JCL for extracting log records from a dump

```
//**********************************************************
//* Extract log records from dump buffers.
//**********************************************************
//IPCSDMP  EXEC PGM=IKJEFT01,REGION=8M
//STEPLIB   DD DSN=IMS.SDFSRESL,DISP=SHR
//*** DATA SET TO WRITE LOGS TO (OUTPUT)
//LOGFILE   DD DISP=(NEW,CATLG),DSN=user.LOGFILE,
//             UNIT=SYSDA,SPACE=(TRK,(5,5),RLSE),
//             RECFM=VB,BLKSIZE=32760,LRECL=32756,BUFNO=3
//*** DATA SET TO WRITE REPORT INFORMATION (OUTPUT)
//RPTFILE   DD DISP=(NEW,CATLG),DSN=user.RPTFILE,
//             UNIT=SYSDA,SPACE=(TRK,(5,5),RLSE),
//             RECFM=FB,LRECL=133,BLKSIZE=1330
//SYSTSPRT  DD SYSOUT=*
//IPCSPRNT  DD SYSOUT=*
//INDEX     DD SYSOUT=*
//SYSABEND  DD SYSOUT=*
//IPCSPARM  DD DISP=SHR,DSN=user.PARMLIB
//          DD DISP=SHR,DSN=SYS1.PARMLIB
//IPCSDDIR  DD DISP=SHR,DSN=user.ISCSDDIR
//INFILE    DD DISP=SHR,DSN=dump.data.set.name
//SYSTSIN   DD *
IPCS NOPARM
SETDEF DSN('dump_data_set_name') +
  NOPROBLEM PRINT NOTERMINAL
VERBX DFSEDA00 'J=IMS2 O(LOGEXTR())'
/*
```

2.2.5 Output log data set example

The output data set will contain log records that can be used in log formatting tools. Figure 2-9 shows part of the output data set containing the log records.

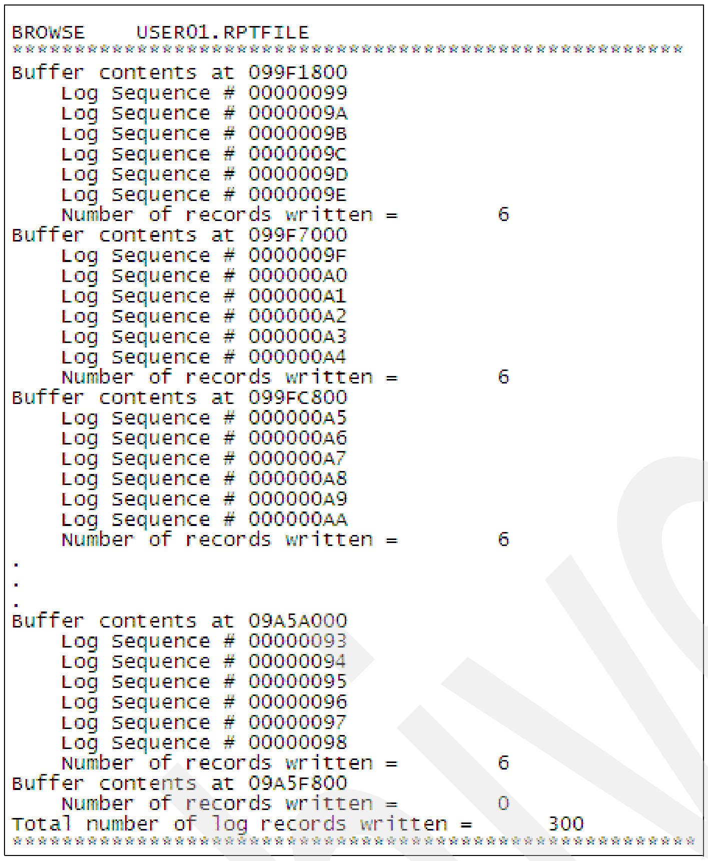


```
BROWSE    USER01.TESTLOG                                               Line 00000000 Col 001 127
********************************************************* Top of Data **********************************************************
......½...c...½.....................................a.d....SHA2     ç ......Ä.æ.............ðôÇ.................ØÀ.....PMA
......½...c...½.....................................a.d....SH2X0    ç ......Ä.æ.............ðõ\.................Ø?°....PM2
......½...c...½.....................................a.d....SMQFP4B açà......Ä.æ.an^.........ð\..................Ø#.....SMQ
......½...c...½.....................................a.d....SMQ3A   A  ......Ä.æ.............ðSH.................Øf°....SMQ
......½...c...½.....................................a.d....TACPCNV1 H ......Ä.æ.H...........ð²ì.................Øk.....TAC
Ç..............................................................................................................................
......½...c...½.....................................a.d....TELEDISP.ç ...x..Ä.æ.............ðÔ½.................Ø¸°....TEL
......½...c...½.....................................a.d....THI23   aç ...-....4.............ðÒø.4...............Øż.....TH1
......½...c...½.....................................a.d....THI54   aç ...-....4.............ðÓÌ.4...............Ø©°....TH1
......½...c...½.....................................a.d....THU33   aç ...-....4.............ðOh.4...............Ø{.....TH1
......½...c...½.....................................a.d....TPCCNO  .ç ...x..Ä.æ.............ðRQ.................ØÔ°....PSB
Ç.......................²:...Ù..¹'......ád..¨................................Ó.......................6.......Ó.......°...
```

Figure 2-9 Output log data set example

2.2.6 Report data set example

The report data set contains log sequence numbers from the log records in the dump buffers. Also included at the bottom of the report is the number of log records written to the data set. Figure 2-10 on page 31 shows the output from the report data set.



```
BROWSE    USER01.RPTFILE
*****************************************************
Buffer contents at 099F1800
    Log Sequence # 00000099
    Log Sequence # 0000009A
    Log Sequence # 0000009B
    Log Sequence # 0000009C
    Log Sequence # 0000009D
    Log Sequence # 0000009E
    Number of records written =        6
Buffer contents at 099F7000
    Log Sequence # 0000009F
    Log Sequence # 000000A0
    Log Sequence # 000000A1
    Log Sequence # 000000A2
    Log Sequence # 000000A3
    Log Sequence # 000000A4
    Number of records written =        6
Buffer contents at 099FC800
    Log Sequence # 000000A5
    Log Sequence # 000000A6
    Log Sequence # 000000A7
    Log Sequence # 000000A8
    Log Sequence # 000000A9
    Log Sequence # 000000AA
    Number of records written =        6
.
.
.
Buffer contents at 09A5A000
    Log Sequence # 00000093
    Log Sequence # 00000094
    Log Sequence # 00000095
    Log Sequence # 00000096
    Log Sequence # 00000097
    Log Sequence # 00000098
    Number of records written =        6
Buffer contents at 09A5F800
    Number of records written =        0
Total number of log records written =      300
*****************************************************
```

Figure 2-10 Report data set example

2.3 Installation verification procedure

The installation verification program (IVP) is provided by IBM to test the product installation of IMS and verify that the major functions and features are working.

The jobs and tasks of the IVP build a sample IMS system and provide several sample applications that are used to verify specific components of IMS.

Use the IVP to verify that IMS has been installed properly and that the major functions and features of IMS are working. For more information about the IVP, refer to *IMS Version 11 Installation*, GC19-2438.

2.3.1 Overview of IVP enhancements

The IVP changes with each IMS version. We discuss what is new with IVP in IMS 11, but also mention the changes in IMS 10 for those who are migrating directly to IMS 11 from IMS 9.

IMS 11

New sample applications are delivered with the IMS IVP. The O series of steps include the jobs and tasks that you must perform during the execution of the Common Service Layer sample application. These steps are updated to include the new Open Database Manager (ODBM) address space.

The general variables are updated.

For more information about IVP changes in IMS 11, refer to 7.4.10, "Changes to the IVP for IMS 11" on page 255.

IMS 10

If you are migrating from IMS 9, many new IVPs are available, including several for functions that were already in IMS 9. IVP changes include:

- OM SPOC Audit Trail display
- RACF implementation
- IMS Java dependent region configuration verification
- IMS Connect sample application
- XQuery sample application
- Dynamic resource definition
- RECON jobs
- IVP Variable Export utility
- Parallel RECON access
- Synchronous Callout function
- Asynchronous Callout function

For more information about IVP changes in IMS 10, refer to *IBM IMS Version 10 Implementation Guide: A Technical Overview*, SG24-7526.

2.4 Syntax Checker

The Syntax Checker is an ISPF application that helps you define, verify, and validate parameters in the members of the PROCLIB data set. It saves the parameters to appropriate PROCLIB members in the correct format. Online help is available at the parameter level and provides assistance with moving to a new IMS release.

2.4.1 Overview of Syntax Checker enhancements

The IMS Syntax Checker is updated for IMS 11 in the following ways:

- Support is added for the following PROCLIB members that are new in IMS 11:

CSLDIxxx	ODBM initialization member
CSLDCxxx	ODBM configuration member
DSPBIxxx	DBRC initialization member

- The Syntax Checker ISPF application is updated in the following ways:
  - The IMS PROCLIB members parameters that were supported in IMS 10 are updated to reflect the changes in IMS 11
  - IMS PROCLIB member parameters that are specific to IMS 8 are no longer supported
- In previous releases, the Syntax Checker did not display all of the data in a selected section. Instead, you had to select the **Expand all** action under the View menu to see all data. IMS 11 changes this. IMS 11 automatically expands selected sections at start time.

Note: The IMS 11 Syntax Checker has no support for IMS 8 PROCLIB members.

For complete information about the Syntax Checker, refer to *IMS Version 11 System Definition,* GC19-2444.

2.4.2 PROCLIB members

Table 2-9 lists the members of the PROCLIB data set that Syntax Checker supports, along with the version for that support. The changes with IMS 11 are in bold font.

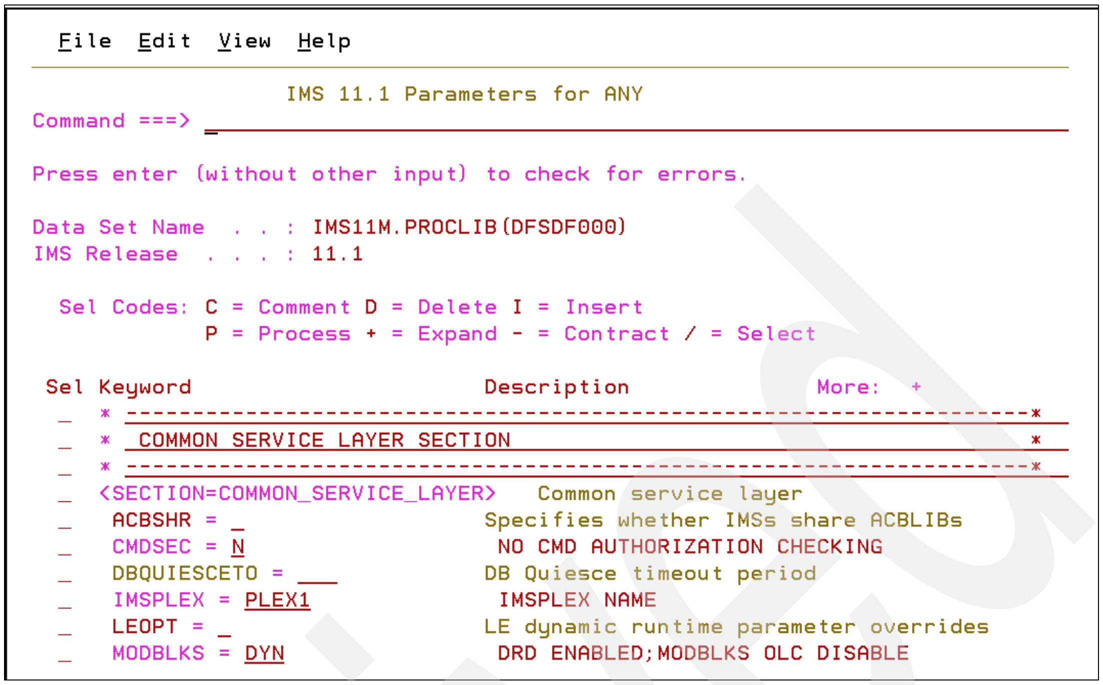
Table 2-9 PROCLIB members supported by the Syntax Checker

PROCLIB member	IMS 9	IMS 10	IMS 11
BPE exit list member (but not BPE configuration members)	yes	yes	yes
CSLDCxxx (ODBM configuration member)	no	no	**yes**
CSLDIxxx (ODBM initialization member)	no	no	**yes**
CSLOIxxx (OM Initialization)	yes	yes	yes
CSLRIxxx (RM initialization)	yes	yes	yes
CSLSIxxx (SCI Initialization)	yes	yes	yes
CQSIPxxx (CQS Initialization)	yes	yes	yes
CQSSGxxx (Global Structure Definition)	yes	yes	yes
CQSSLxxx (CQS Local Structure Definition)	yes	yes	yes
DFSCGxxx (Common Service Layer)	yes	yes	yes
DFSDCxxx (IMS data communication options)	yes	yes	yes
DFSDFxxx (Dynamic Resource Definition)	no	yes	yes
DFSPBxxx (IMS execution parameters)	yes	yes	yes
DFSSQxxx (Shared Queues)	yes	yes	yes
DSPBIxxx (DBRC Initialization member)	no	no	**yes**
HWSCFGxx (IMS Connect Configuration member)	yes	yes	yes

2.4.3 Keyword Display panel

The Syntax Checker Keyword Display panel, shown in Figure 2-11 on page 34, displays the keywords and their values and indicates whether syntax errors exist.

In previous releases, the Syntax Checker did not display all of the data in a selected section. Instead, the user had to select the **Expand all** action under the View menu to see all data. Now, IMS 11 automatically expands selected sections at start time.



```
  File  Edit  View  Help
                   IMS 11.1 Parameters for ANY
Command ===> _
Press enter (without other input) to check for errors.

Data Set Name  . . : IMS11M.PROCLIB(DFSDF000)
IMS Release  . . . : 11.1

  Sel Codes: C = Comment D = Delete I = Insert
             P = Process + = Expand - = Contract / = Select

 Sel Keyword                      Description              More:   +
  _  * --------------------------------------------------------------------*
  _  *  COMMON SERVICE LAYER SECTION                                       *
  _  * --------------------------------------------------------------------*
  _  <SECTION=COMMON_SERVICE_LAYER>    Common service layer
  _   ACBSHR = _                  Specifies whether IMSs share ACBLIBs
  _   CMDSEC = N                   NO CMD AUTHORIZATION CHECKING
  _   DBQUIESCETO = ___           DB Quiesce timeout period
  _   IMSPLEX = PLEX1              IMSPLEX NAME
  _   LEOPT = _                   LE dynamic runtime parameter overrides
  _   MODBLKS = DYN                DRD ENABLED;MODBLKS OLC DISABLE
```

Figure 2-11 Keyword Display panel

2.4.4 Display options for the Keyword Display panel

The Syntax Checker display options that you can select from the View menu on the action bar are:

- Select Display

  Display all keywords with values. These keywords are saved when the member is saved.

- All Display

  Display all possible keywords. Only keywords with values are saved when the member is saved.

- New Display

  Display all new keywords in this release of IMS. These keywords are saved only if they have values when the member is saved.

The status of the member determines whether keywords are displayed the first time the panel is displayed.

If the member is new or empty, then a list of all possible keywords for the member is displayed (All Display).

If the member is not empty or new, then the list contains the current keywords defined in the member (Select Display). You can obtain a display of all possible keywords for the member by using the **View** → **Display All** option.

To display the default values for parameters, press the F6 function key to toggle between displaying and not displaying defaults. The default values are displayed in the description field of the parameter.

2.4.5 Syntax Checker error checking

After modifying the member, press Enter without making any other modifications. Syntax value checking is performed, as follows:

- ► If errors exist, the first keyword with an error moves to the top of the display and an error message is displayed. The error must be corrected before the next error can be displayed.
- ► When all errors are resolved and you have modified the member as required, you can save the member to the originally selected PROCLIB and member or to a different PROCLIB and member.

2.4.6 Action pull-down options on the Keyword Display panel

Action pull-down options are located at the top of the Keyword Display panel. They are File, Edit, and View.

File action options

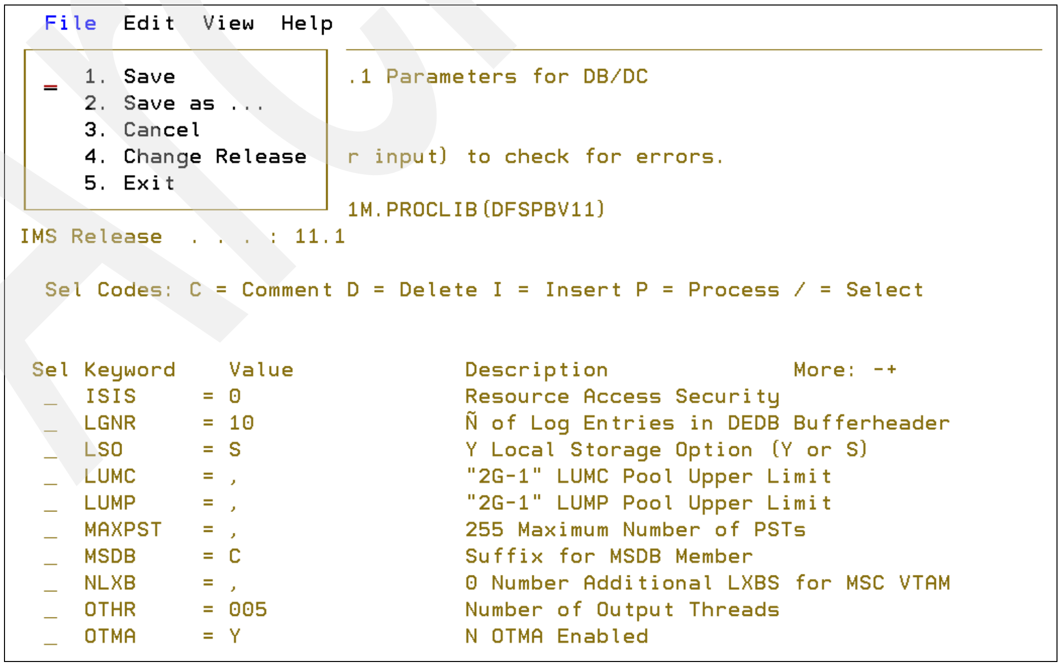
The File action options (shown in Figure 2-12) are:

- ► Save

  The keywords are stored back in the originally selected PROCLIB member. You are notified of any problems before the Syntax Checker saves the member, and you can either save or not save the member.
- ► Save As

  The keywords are stored to a specific data set and member. You are prompted for the data set name and member name.
- ► Change Release

  This option changes the IMS release of the parameter member being processed, by giving an option to save the member before changing the release and reprocessing the member under the new release, to identity any value errors and invalid or obsolete keywords.



```
  File   Edit   View   Help
                            ________________________________________________
  _   1. Save              .1 Parameters for DB/DC
      2. Save as ...
      3. Cancel
      4. Change Release    r input) to check for errors.
      5. Exit
                           1M.PROCLIB(DFSPBV11)
 IMS Release  . . . : 11.1

  Sel Codes: C = Comment D = Delete I = Insert P = Process / = Select


 Sel Keyword     Value            Description                 More: -+
  _  ISIS      = 0                Resource Access Security
  _  LGNR      = 10               Ñ of Log Entries in DEDB Bufferheader
  _  LSO       = S                Y Local Storage Option (Y or S)
  _  LUMC      = ,                "2G-1" LUMC Pool Upper Limit
  _  LUMP      = ,                "2G-1" LUMP Pool Upper Limit
  _  MAXPST    = ,                255 Maximum Number of PSTs
  _  MSDB      = C                Suffix for MSDB Member
  _  NLXB      = ,                0 Number Additional LXBS for MSC VTAM
  _  OTHR      = 005              Number of Output Threads
  _  OTMA      = Y                N OTMA Enabled
```

Figure 2-12 File action options

Edit action options

The Edit action options (shown in Figure 2-13) are:

- Comments

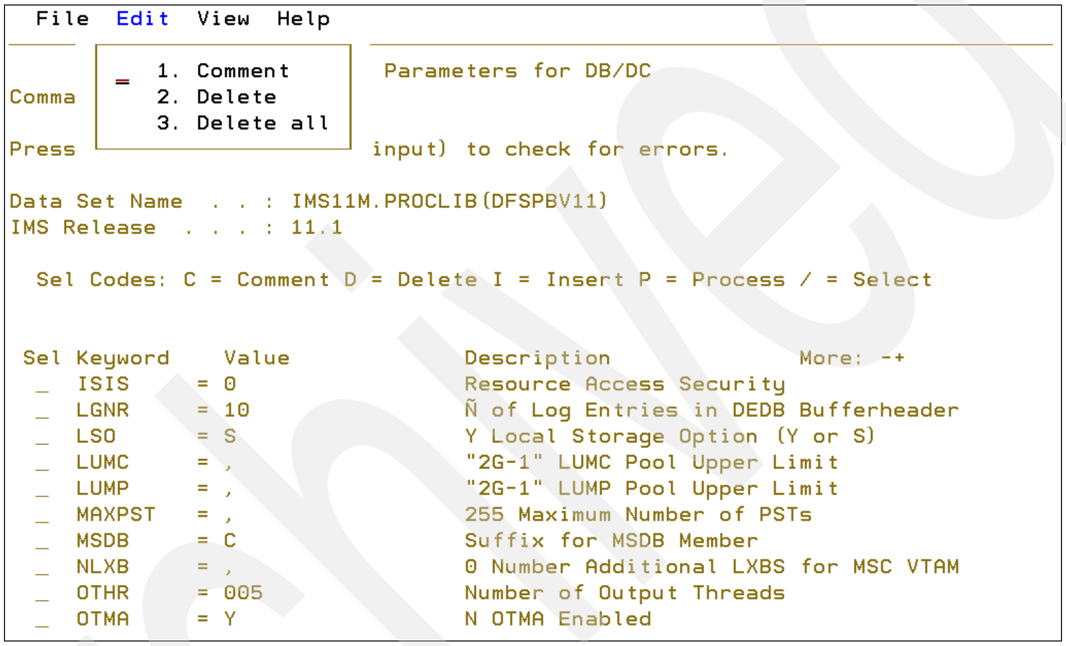
  Adds a comment to keywords that have been selected with the forward slash (/) operand.

- Delete

  Deletes keywords that have been selected with the forward slash (/) operand.

- Delete All

  Deletes all keywords from the display.



```
  File  Edit  View  Help
 ______ ___________________ ___________________________________________________
       |                   |
       | _  1. Comment     | Parameters for DB/DC
 Comma |    2. Delete      |
       |    3. Delete all  |
 Press |___________________| input) to check for errors.

 Data Set Name  . . : IMS11M.PROCLIB(DFSPBV11)
 IMS Release  . . . : 11.1

   Sel Codes: C = Comment D = Delete I = Insert P = Process / = Select


  Sel Keyword    Value              Description              More:  -+
   _   ISIS     = 0                 Resource Access Security
   _   LGNR     = 10                Ñ of Log Entries in DEDB Bufferheader
   _   LSO      = S                 Y Local Storage Option (Y or S)
   _   LUMC     = ,                 "2G-1" LUMC Pool Upper Limit
   _   LUMP     = ,                 "2G-1" LUMP Pool Upper Limit
   _   MAXPST   = ,                 255 Maximum Number of PSTs
   _   MSDB     = C                 Suffix for MSDB Member
   _   NLXB     = ,                 0 Number Additional LXBS for MSC VTAM
   _   OTHR     = 005               Number of Output Threads
   _   OTMA     = Y                 N OTMA Enabled
```

Figure 2-13 Edit action options

View action options

The View action options (shown in Figure 2-14 on page 37) are:

- Display All

  Changes the keyword display to include all possible keywords for the selected IMS release and control region type.

- Display Selected

  Changes the keyword display to include only the keywords that have a value.

- Display New

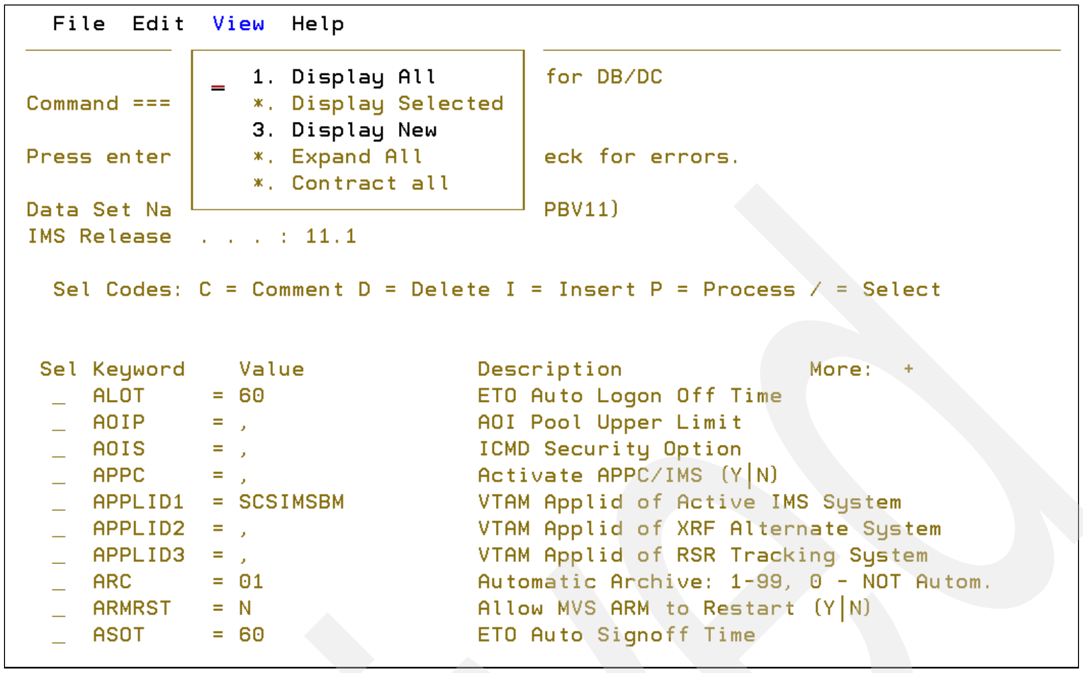
  Changes the keyword display to include only the keywords that are new for the IMS release.

- + (Expand)

  Displays the keyword on multiple lines.

- - (Contract)

  Displays the keywords on one line.

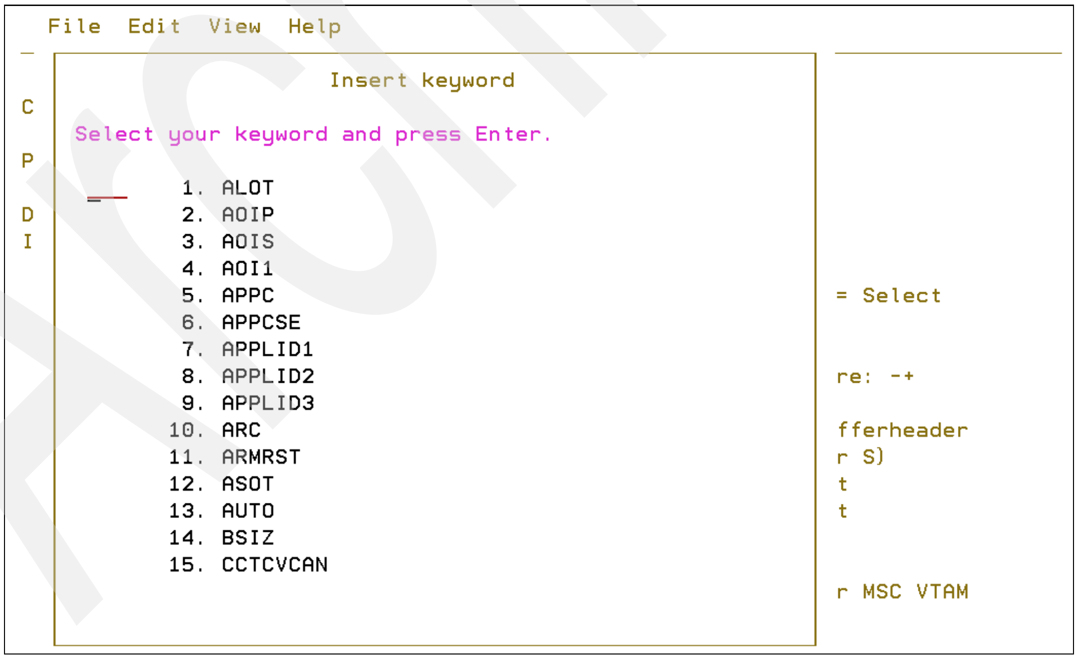


```
  File   Edit   View   Help
 ___________  ________________________  ________________________________________
              _   1. Display All         for DB/DC
Command ===       *. Display Selected
                  3. Display New
Press enter       *. Expand All          eck for errors.
                  *. Contract all
Data Set Na                              PBV11)
IMS Release   . . . : 11.1

  Sel Codes: C = Comment D = Delete I = Insert P = Process / = Select


 Sel Keyword    Value              Description              More:   +
  _  ALOT     = 60                 ETO Auto Logon Off Time
  _  AOIP     = ,                  AOI Pool Upper Limit
  _  AOIS     = ,                  ICMD Security Option
  _  APPC     = ,                  Activate APPC/IMS (Y|N)
  _  APPLID1  = SCSIMSBM           VTAM Applid of Active IMS System
  _  APPLID2  = ,                  VTAM Applid of XRF Alternate System
  _  APPLID3  = ,                  VTAM Applid of RSR Tracking System
  _  ARC      = 01                 Automatic Archive: 1-99, 0 - NOT Autom.
  _  ARMRST   = N                  Allow MVS ARM to Restart (Y|N)
  _  ASOT     = 60                 ETO Auto Signoff Time
```

Figure 2-14 View action options

2.4.7 Inserting keywords into the Keyword Display panel

To insert a keyword into the display, type an **I** (insert) in the Sel field and press Enter. Figure 2-15 shows a window that opens and lists the possible items that can be inserted.



```
  File   Edit   View   Help
 _                                                        _________________
                         Insert keyword
C
   Select your keyword and press Enter.
P
      __          1. ALOT
D                 2. AOIP
I                 3. AOIS
                  4. AOI1
                  5. APPC                                 = Select
                  6. APPCSE
                  7. APPLID1
                  8. APPLID2                              re: -+
                  9. APPLID3
                 10. ARC                                  fferheader
                 11. ARMRST                               r S)
                 12. ASOT                                 t
                 13. AUTO                                 t
                 14. BSIZ
                 15. CCTCVCAN
                                                          r MSC VTAM
```

Figure 2-15 Insert Keyword panel

2.4.8 Saving processed members Keyword Display panel

While using the Keyword Display Panel, the processed member can be saved to the originally selected PROCLIB and member or to a different PROCLIB and member by using one of the Save or Save As options, as follows:

- The parameters are stored in alphabetical order.
- Only the parameters that are displayed in the Display Selected keyword panel are stored.

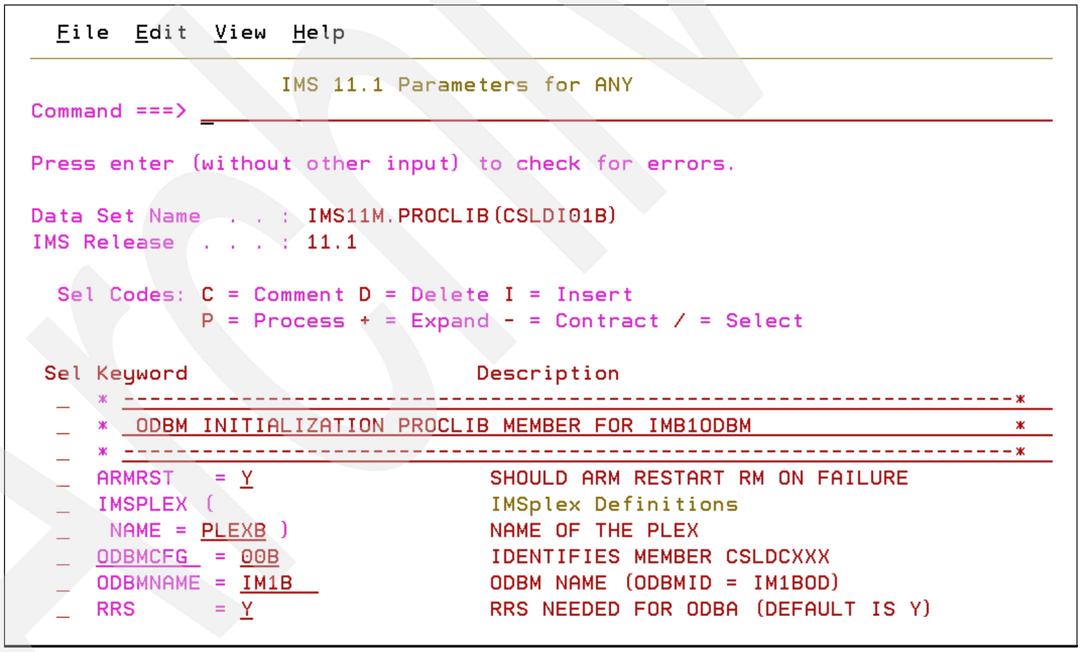
For members that support comments, they are stored in front of their keyword and the Syntax Checker adds comments to the top of the member as shown in Example 2-6.

Example 2-6 Informational comments

```
*<VERSION>11.1
*<IMSCR>DB/DC
*<DATE>09/07/29
*<TIME>20:04
```

2.4.9 CSLDIxxx ODBM initialization member

The IMS 11 Syntax Checker supports the new CSLDIxxx ODBM initialization member. Figure 2-16 shows a sample of this panel.



```
  File  Edit  View  Help
                   IMS 11.1 Parameters for ANY
Command ===> _

Press enter (without other input) to check for errors.

Data Set Name  . . : IMS11M.PROCLIB(CSLDI01B)
IMS Release  . . . : 11.1

  Sel Codes: C = Comment D = Delete I = Insert
             P = Process + = Expand - = Contract / = Select

 Sel Keyword                      Description
  _  * -------------------------------------------------------------------*
  _  *  ODBM INITIALIZATION PROCLIB MEMBER FOR IMB1ODBM                   *
  _  * -------------------------------------------------------------------*
  _  ARMRST    = Y                 SHOULD ARM RESTART RM ON FAILURE
  _  IMSPLEX (                     IMSplex Definitions
  _   NAME = PLEXB )               NAME OF THE PLEX
  _  ODBMCFG   = 00B               IDENTIFIES MEMBER CSLDCXXX
  _  ODBMNAME  = IM1B              ODBM NAME (ODBMID = IM1BOD)
  _  RRS       = Y                 RRS NEEDED FOR ODBA (DEFAULT IS Y)
```

Figure 2-16 ODBM Initialization Member panel

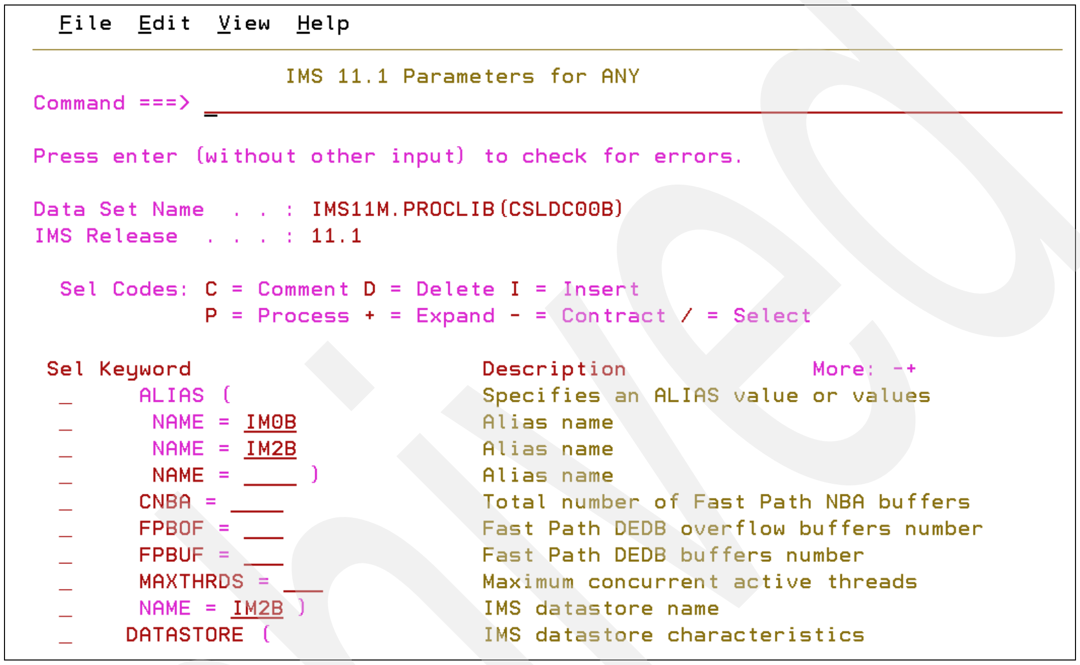
This PROCLIB member specifies parameters that relate to initializing the ODBM address space, such as:

- Whether the z/OS Automatic Restart Manager (ARM) is used to restart the ODBM address space after an abend
- Whether ODBM registers with z/OS Recovery Resource Services (RRS) and uses the ODBA API or does not register with RRS and uses the DRA API instead
- The name for the ODBM address space

- The suffix for the ODBM configuration PROCLIB member
- The IMSplex name

2.4.10 CSLDCxxx ODBM configuration member

The IMS 11 Syntax Checker supports the new CSLDCxxx ODBM configuration member. Figure 2-17 shows a sample of this panel.



```
  File  Edit  View  Help
                   IMS 11.1 Parameters for ANY
Command ===> _

Press enter (without other input) to check for errors.

Data Set Name  . . : IMS11M.PROCLIB(CSLDC00B)
IMS Release  . . . : 11.1

  Sel Codes: C = Comment D = Delete I = Insert
             P = Process + = Expand - = Contract / = Select

 Sel Keyword                         Description              More:  -+
  _      ALIAS (                     Specifies an ALIAS value or values
  _       NAME = IM0B                Alias name
  _       NAME = IM2B                Alias name
  _       NAME = ____ )              Alias name
  _      CNBA = ____                 Total number of Fast Path NBA buffers
  _      FPBOF = ___                 Fast Path DEDB overflow buffers number
  _      FPBUF = ___                 Fast Path DEDB buffers number
  _      MAXTHRDS = ___              Maximum concurrent active threads
  _      NAME = IM2B )               IMS datastore name
  _     DATASTORE (                  IMS datastore characteristics
```

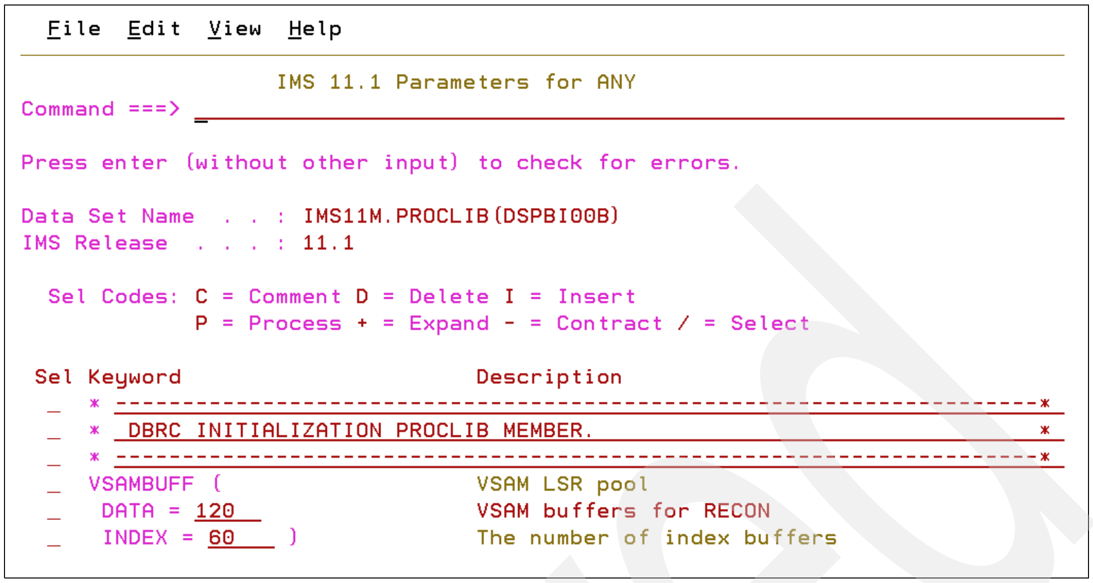
Figure 2-17 ODBM Configuration Member panel

This PROCLIB member specifies the data store initialization parameters, which are either global or local:

- The global parameters specify:
  - The number of times ODBM will attempt to connect to an IMS data store
  - The time (in seconds) between attempts by ODBM to establish a connection to an IMS data store
  - The number of concurrent active threads
  - The number of Fast Path normal buffer allocation (NBA) and overflow (OBA) buffers
- The local parameters specify:
  - The name of the ODBM address space
  - The name of the IMS data store
  - The name of the alias for the IMS data store
  - Other optional parameters that override certain global parameters

2.4.11 DSPBIxxx DBRC initialization member

The IMS 11 Syntax Checker supports the new DSPBIxxx DBRC initialization member. Figure 2-18 on page 40 shows a sample of the panel.



```
  File  Edit  View  Help
                   IMS 11.1 Parameters for ANY
Command ===> _
Press enter (without other input) to check for errors.
Data Set Name  . . : IMS11M.PROCLIB(DSPBI00B)
IMS Release  . . . : 11.1

  Sel Codes: C = Comment D = Delete I = Insert
             P = Process + = Expand - = Contract / = Select

 Sel Keyword                         Description
  _  * -------------------------------------------------------------------*
  _  *  DBRC INITIALIZATION PROCLIB MEMBER.                                *
  _  * -------------------------------------------------------------------*
  _  VSAMBUFF (                      VSAM LSR pool
  _   DATA = 120                     VSAM buffers for RECON
  _   INDEX = 60    )                The number of index buffers
```

Figure 2-18 DBRC Initialization Member panel

Use the DSPBIxxx PROCLIB member to specify parameters that initialize the DBRC address space:

IMSPLEX()	Specifies the IMSplex name used by DBRC for SCI registration.
DBRCGRP=	Specifies the three-character DBRC group ID within the IMSplex.
VSAMBUFF()	Specifies the maximum number of index and data buffers to be assigned to the VSAM LSR pool.

2.5 LSQA storage reduction

The IMS main internal storage managing service, IMODULE, uses a z/OS control block structure (CDEs) to keep track of areas of storage that are obtained through the IMODULE GETMAIN function.

These contents directory entries (CDEs) are then used later when the storage is deleted (IMODULE DELETE) to verify that the storage is indeed allocated. The CDEs are also used to free all storage at IMS address space termination.

The IMS 11 IMODULE storage manager service is updated to use 64-bit private storage, instead of 24-bit authorized private storage for certain IMS functions.

This approach is to support the optional tracking of IMODULE GETMAINed storage using an IMS-defined tracking structure, instead of the CDE structure defined by z/OS.

Note: Be aware of this change if you currently scan CDEs to find a particular piece of storage.

A glossary of terminology used in the LSQA storage reduction enhancements is in 2.5.9, “Glossary of LSQA terms” on page 49.

2.5.1 Overview

IMS provides an internal IMS service called IMODULE, which the IMS modules use to allocate and release storage, and load and delete modules. IMODULE keeps track of both storage areas and modules by building a control block structure that is defined by z/OS (CDEs and related blocks).

These z/OS blocks must architecturally reside in 24-bit authorized-private storage (LSQA storage). The 24-bit private storage is a limited resource (limited to a maximum of 16 MB, but more practically in the range of 8 - 10 MB).

Tracking method selection

In IMS 11, there are now two tracking methods available for managing storage obtained by the IMODULE GETMAIN function.

- IMODULE can track storage by CDEs, as it has in prior releases of IMS. This is the default.
- IMODULE can also now trace storage using IMS-defined tracking blocks, which are built in 64-bit private storage.

The selection of a tracking method to use is on a per-GETMAIN basis, through a parameter on the IMODULE macro.

The new 64-bit tracking method is available only for storage obtained by the IMS control region or the IMS DLI region. Other IMS region types (dependent regions, batch regions, DBRC) continue to use CDEs for all of their storage.

End-of-memory IMS abends reduction

This enhancement should reduce the occurrences of end-of-memory (EOM) type IMS abends that, to be resolved, require an IPL of the z/OS system.

With the large size of today’s address spaces, it is possible to allocate more storage areas in 31-bit storage than is possible to track by using 24-bit CDE structures.

When the allocation exceeds the tracking in the IMS CTL or DLI address spaces, often is the case that z/OS itself cannot get enough storage to perform recovery/termination manager (RTM) processing for the address space.

This problem leads to EOM type abends, where IMS is unable to clean up its allocated common storage.

This inability to clean up often requires a z/OS IPL to clear the *orphaned* common storage so that IMS can be restarted on that z/OS.

2.5.2 A 32-bit storage memory map

Figure 2-19 on page 42 shows a 31-bit (2 GB) storage memory map. The 2 GB of storage is divided into many storage areas.

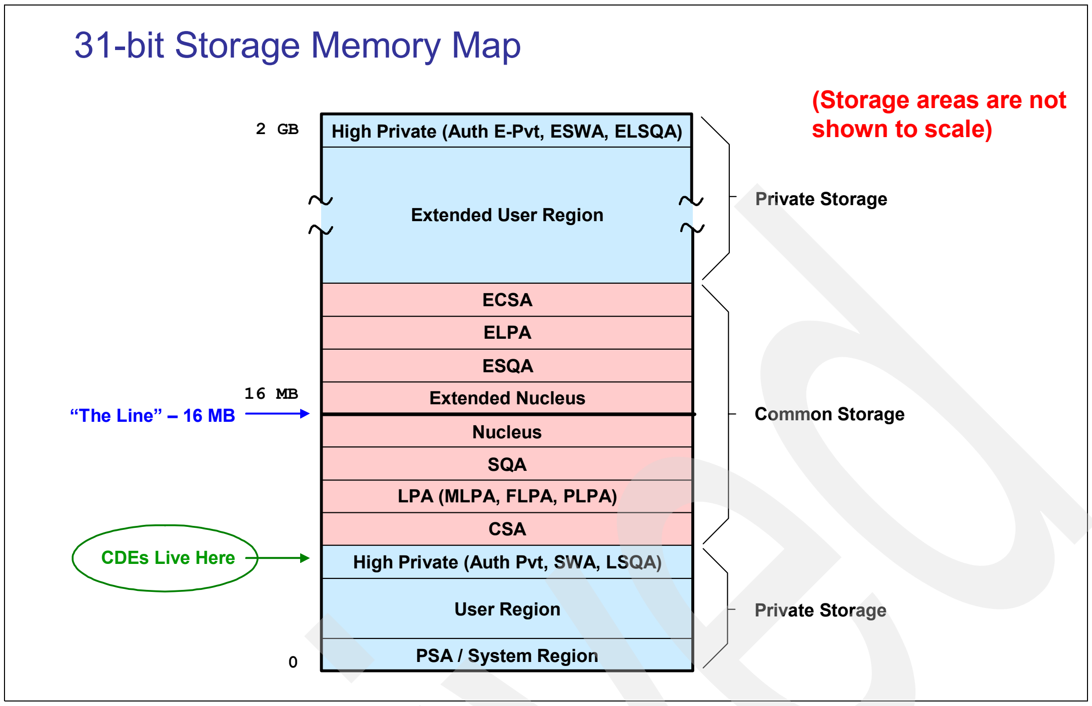


Figure 2-19 31-bit storage memory map

At the highest level, storage begins and ends with private storage. In the middle (surrounding the *16 MB line*) is a region of common storage:

- *Private storage* is private or local to each address space. That is, two programs, which both refer to the same address within the private storage range, refer to two separate pieces of storage.
- *Common storage* is shared among all address spaces. Two programs, which both refer to the same address within the common storage range, refer to the same piece of storage.

Each private area (24-bit private and 31-bit extended private) is divided into two major sections:

- *User region storage* is allocated from the bottom of each private area up. User region is the storage that is available to be allocated by any program running in an address space, whether or not the program is authorized. The amount of user region storage can be limited by the RGN= parameter on a job, or by SMF exits.
- *High private* contains various authorized private storage subpools, and can only be allocated by an authorized program.

Common storage is divided into several sections. The order of the sections is mirrored around the 16 MB line:

- Below the line, common storage is allocated as CSA, LPA, SQA, and Nucleus.
- Above the line, common storage is allocated in the reverse order: Extended Nucleus, ESQA, ELPA, ECSA.

AMODE 24 programs can only access storage below *the line* at 16 MB. The reason is because 24 bits of storage give an addressing range of 0 - 16 MB (AMODE 24 programs cannot see anything beyond what 24 bits can address).

AMODE 31 programs can access storage both above and below the line.

CDEs are allocated by z/OS and IMS in the 24-bit High Private area (LSQA). Because it has to coexist with the other 24-bit areas, it is a limited resource and can easily be exhausted.

2.5.3 LSQA reduction scenario

CDEs are allocated in 24-bit authorized-private storage (LSQA). Each CDE and its related blocks occupy 64 bytes in 24-bit private storage.

LSQA storage is allocated from the top of the 24-bit private region downward, toward the non-authorized, user-private region. LSQA is not subject to region limits.

OTMA flood

In cases where IMS allocates large numbers of relatively small storage areas, completely filling the 24-bit private area with CDEs is possible. A common scenario for this is the OTMA flood case, where a certain condition prevents IMS from being able to process OTMA input messages.

IMS uses CDEs to map its ITASK blocks (YTIB and YQAB). OTMA flood conditions can lead to the creation of large numbers of OTMA ITASKs, consuming a significant amount of storage.

The YTIBs and associated blocks are not built in LSQA. However, what is built in LSQA are the z/OS CDEs for these blocks. IMS storage management for these types of blocks creates CDEs in LSQA to map the size and location of the actual block.

IMS treats those blocks as load modules and follows the rules for CDEs, building them in LSQA below the line subpools, although the actual blocks can in fact be in high-private, above the line. The number of blocks built can be so large that too much LSQA may be consumed simply in mapping them, causing the RTM failures / S40D / Terminated at EOM problems.

Orphaned common storage

When 24-bit storage is totally exhausted, often the case is that z/OS is unable to obtain the storage that it needs to do the processing to terminate the address space cleanly. This occurrence most often manifests itself as a z/OS ABEND40D.
When this happens, z/OS skips address-space-related cleanup, and goes directly to end-of-memory (EOM) processing. No IMS cleanup routines are called at the task level. The address space is terminated and the only cleanup that is called is EOM cleanup.

EOM cleanup runs after the address space is gone, under the z/OS Master Scheduler address space. EOM processing does not have access to the CDEs that were in the IMS LSQA storage, because by the time EOM runs, that storage is gone.

Thus, EOM cleanup is not able to free common storage that IMS had tracked through the CDEs, leaving *orphaned* common storage in the z/OS image, and very often leading to the customer having to re-IPL the system in order to be able to start IMS.

IMS BCB IPAGE storage expansion

The IMODULE GETMAIN storage requests for an internal type of IMS block (called a BCB IPAGE) are changed to request tracking by IMS STEs, rather than by z/OS CDEs.

BCB IPAGE storage is heavily used for many IMS internal processes and control block structures. Often, run-away conditions lead to the allocation of many IPAGEs of storage, and can lead to the out-of-storage and end-of-memory conditions.

A certain situation develops, which causes a rapid expansion in IMS BCB IPAGE storage. For example, a problem could exist with the processing of IMS messages, leading to queuing of messages and the concomitant allocation of control blocks to represent the messages, as follows:

1. IMS allocates IPAGEs, typically in 31-bit storage. Each IPAGE is tracked by a z/OS CDE structure, allocated in 24-bit authorized private storage (LSQA).
2. All available LSQA storage is consumed by IMS-built CDE blocks before 31-bit storage is filled.
3. IMS abends (typically, an 878 out-of-storage abend, although other abends are possible).
4. z/OS RTM cleanup runs and attempts to get 24-bit LSQA storage. No storage is available.
5. z/OS terminates the address space with ABEND40D, resulting in IMS cleanup being skipped. The common storage that IMS had allocated is stranded.
6. Restarting of IMS fails because not enough remaining common storage is on the LPAR to allow IMS to start.
7. z/OS is IPLed to recover from the loss of the common storage, resulting in a substantially elongated outage.

2.5.4 IMODULE storage tracking before IMS 11

Before IMS 11, all storage areas obtained by either IMODULE LOAD or IMODULE GETMAIN are tracked by DFSMODU0 building a block structure, referred to as a contents directory entry (CDE).

The CDEs are queued off of contents-manager chains in whatever TCB is assigned ownership of the storage (in IMS, this is usually the job step TCB (STM or DLI). CDEs reside in 24-bit LSQA storage, and have to be there architecturally. The 24-bit storage is a limited resource, even private 24-bit storage. Building CDEs in 24-bit storage is not scalable with the sizes of today's address spaces. Each CDE consumes 64 bytes.

Figure 2-20 on page 45 shows the logical structure of CDE-tracked storage prior to IMS 11. The CDEs shown here are really a collection of blocks; for purposes of illustration, they are shown as one block.

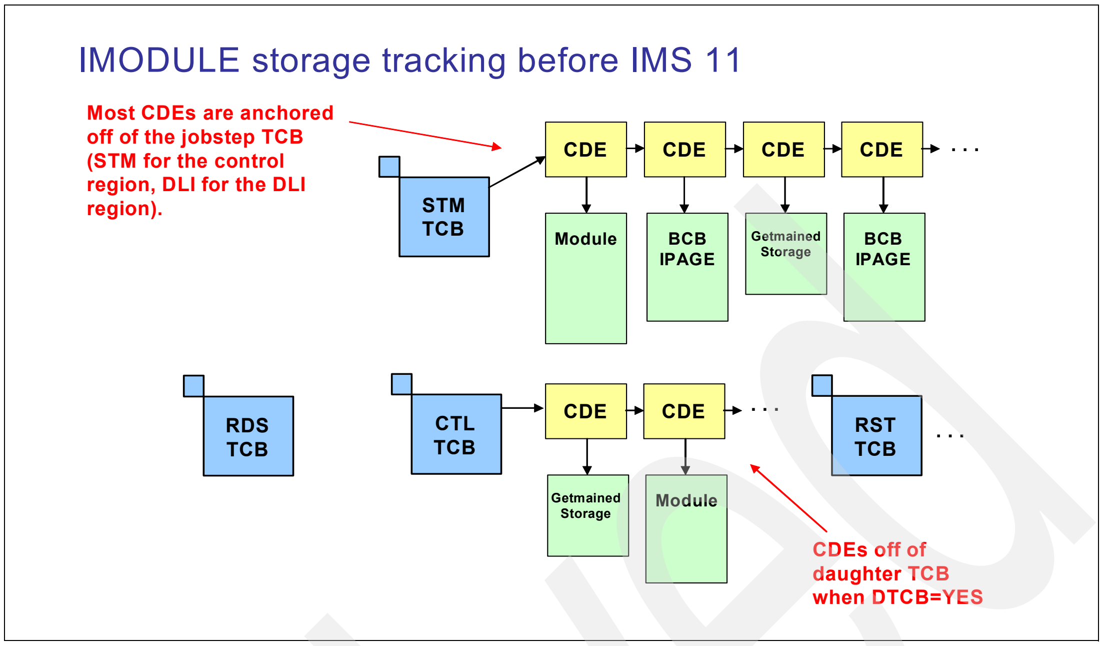

IMODULE storage tracking before IMS 11
Most CDEs are anchored off of the jobstep TCB (STM for the control region, DLI for the DLI region).
STM TCB
CDE
CDE
CDE
CDE
Module
BCB IPAGE
Getmained Storage
BCB IPAGE
RDS TCB
CTL TCB
CDE
CDE
Getmained Storage
Module
RST TCB
CDEs off of daughter TCB when DTCB=YES

Figure 2-20 IMODULE storage tracking before IMS 11

Most CDEs are anchored from the job step TCB (STM TCB for the CTL region; DLI TCB for the DLI region). CDEs can point to a load module (through IMODULE LOAD), or to GETMAINed storage (IMODULE GETMAIN). BCB IPAGEs are a specific type of GETMAINed storage.

Storage can be assigned as being owned by a TCB, other than job step, by coding DTCB=YES on the IMODULE macro. When this approach is done, the current TCB, not the job step TCB is assigned ownership, and the CDE structure is anchored off of the current TCB. Relatively few storage areas are tracked this way in IMS.

2.5.5 A 64-bit storage memory map

A 64-bit virtual storage provides 16 exabytes (EB) in addressable storage range. The new storage is added above the old upper storage limit of 2 GB, now referred to as the *2 GB bar*, or simply *the bar*. Thus, the term *storage above the bar* refers to 64-bit storage.

Figure 2-21 shows this memory map.

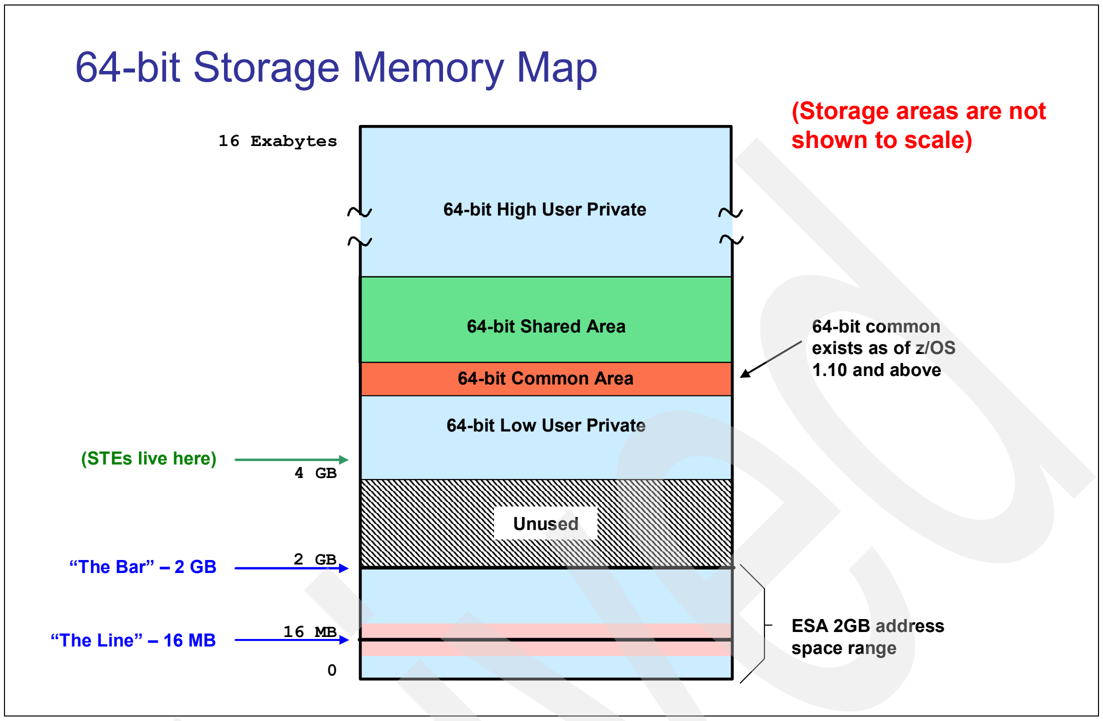


Figure 2-21 64-bit storage memory dump

The implementation of 64-bit storage actually begins above-the-bar storage at 4 GB, rather than immediately above the 2 GB bar.

The storage range of 2 - 4 GB is not used. It corresponds to addresses in the range of X'0000000_80000000' - X'00000000_FFFFFFFF'

Starting at 4 GB (address 00000001_00000000) is 64-bit *low* private storage. It behaves similar to private storage (24- or 31-bit) in that it is local to the address space that allocated it.

With z/OS 1.10 and later, the next storage area is 64-bit common storage, which is visible to all address spaces, as is 24- and 31-bit common storage.

The size of the 64-bit common area is installation-selectable from 2 GB up to 1 TB, in increments of 2 GB. For z/OS 1.9 and earlier, there is no 64-bit common storage.

Above the 64-bit common storage is the 64-bit shared area. Shared storage is similar to common storage in that it can be accessed by multiple address spaces. However, unlike common, shared is only visible to address spaces that *attach* to it.

The remaining storage above the shared area is 64-bit high-private storage. As with the low private storage, the high-private storage is local to the address space that allocates it.

The layout of the below-the-bar 2 GB area remains the same as it was.

2.5.6 LSQA storage reduction

LSQA storage reduction function is enabled in the base IMS 11 code. No user action is required to invoke or enable it.

IMODULE changes

The IMODULE storage manager service is changed to support the tracking of IMODULE GETMAINed storage by using an IMS-defined tracking structure, instead of the z/OS defined CDE structure.

The selection of a tracking method to use is on a per-GETMAIN basis, through a new parameter on the IMODULE macro. The IMS storage tracking elements (STEs) are built in 64-bit private storage for the CTL region and DLI region.

This tracking method is available only for storage obtained by the IMS control region or the IMS DLI region. Other IMS region types (dependent regions, batch regions, DBRC) continue to use CDEs for all of their storage.

IMS BCB IPAGE storage expansion

The IMODULE GETMAIN call that allocates IMS BCB IPAGEs is changed to request the new tracking mechanism, rather than by z/OS CDEs. This approach eliminates the allocation of CDEs for all IPAGE storage.

Similarly, the IMODULE DELETE that deletes IMS BCB IPAGEs is changed to specify the new tracking mechanism.

In a situation that causes a rapid expansion in IMS BCB IPAGE storage, for example, when the processing of IMS messages leads to queuing of messages and the concomitant allocation of control blocks to represent the messages, this is what now happens:

1. IMS allocates IPAGEs, typically in 31-bit storage. Each IPAGE is tracked by an IMS storage tracking element, allocated in 64-bit private storage.
2. All available 31-bit private storage is consumed by the expanding BCB IPAGEs.
3. IMS abends (typically, an 878 out-of-storage abend, although other abends are possible).
4. z/OS RTM cleanup runs and attempts to get 24-bit LSQA storage. Because no CDE structures were built for the IPAGEs (as would have occurred prior to this LSQA enhancement), there is 24-bit storage available.
5. z/OS RTM calls the IMS cleanup routines, which free the allocated common storage of the IMS.
6. The IMS address space terminates.
7. Sufficient common storage is available and IMS restarts successfully. An IMS outage did occur, but the extended outage because of the z/OS IPL is avoided.

LOC=ONLY31 storage

All BCB IPAGEs and DFSPOOL pools in 31-bit storage are now changed to LOC=ONLY31 storage.

This storage attribute was introduced to the IMS storage manager services in IMS 10. It prevents 31-bit requests from *wrapping* down to 24-bit storage when no more 31-bit storage is available.

2.5.7 IMODULE storage tracking in IMS 11

In IMS 11, the BCB IPAGE tracking is moved to the new Storage Tracking Element structure. Figure 2-22 shows that the BCB IPAGEs are moved and are no longer found in the CDE chains off of the TCBs.

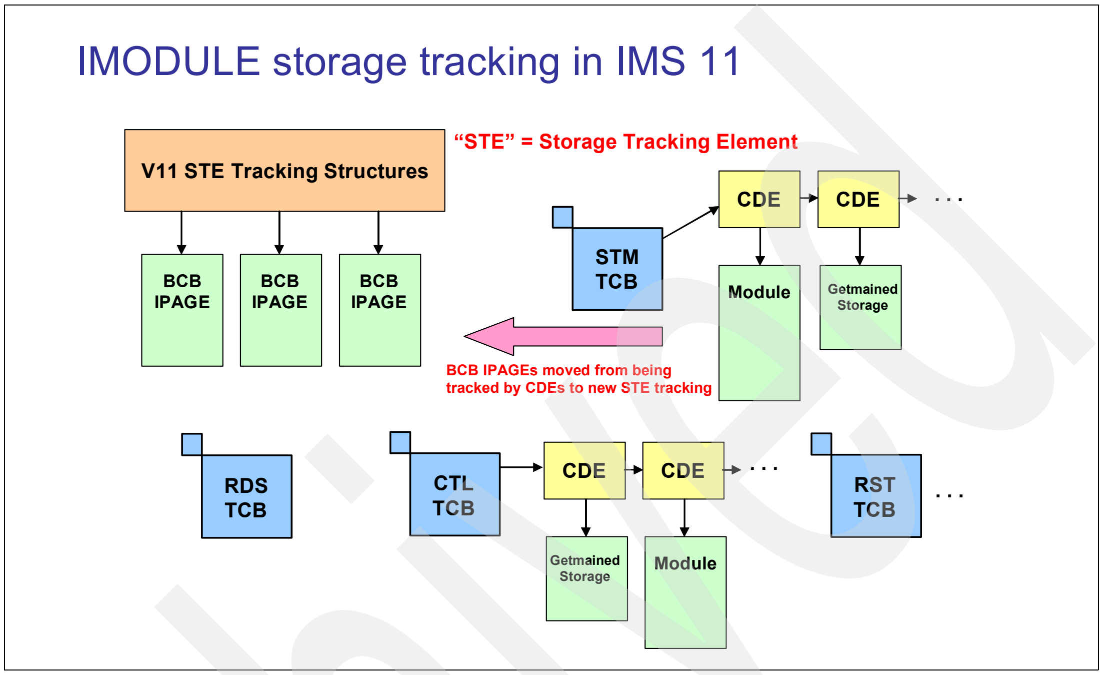

IMODULE storage tracking in IMS 11
V11 STE Tracking Structures
"STE" = Storage Tracking Element
BCB IPAGE
BCB IPAGE
BCB IPAGE
STM TCB
CDE
CDE
Module
Getmained Storage
BCB IPAGEs moved from being tracked by CDEs to new STE tracking
RDS TCB
CTL TCB
CDE
CDE
Getmained Storage
Module
RST TCB

Figure 2-22 IMODULE storage tracking in IMS 11

From IMODULE's point of view, any piece of IMODULE GETMAIN storage can be tracked by the new tracking structure; in IMS 11, only BCB IPAGEs are gotten with the new tracking option requested.

2.5.8 Other IMODULE enhancements

This section discusses several other enhancements.

Statistics

Statistics that are related to the new storage manager tracking are kept and are externalized in new fields in the existing X'4512' log record (IMODULE statistics record).

There are four storage tracking hash tables, and hence four sets of statistics that are in the log record:

- Control region common storage
- Control region private storage
- DLI region common storage
- DLI region private storage

Each area of storage contains the statistics as listed here. This is information only about storage that is tracked by using the new STE approach, not about all storage that is allocated in general.

Diagnostics

Diagnostic information is captured about each area of storage that is tracked using the new IMS storage tracking elements in addition to what is available with contents supervision tracking (CDEs).

New diagnostic information is captured by the IMS storage manager in general. One difficulty in diagnosing end-of-memory abends, such as ABEND40D, is that the IMS control region private storage is not dumped.

This diagnostics information enhancement saves certain storage-related statistics and trace information into a small common storage table, so that this data can be available in dumps, even when IMS private storage is not available.

All new storage tracking structures can be formatted in dumps.

Messages and codes

Figure 2-23 shows the new messages.

```
DFS4342E IMODULE INIT FAILED; RC=xx  RSN=yyyyyyyy
DFS4343E IMS STORAGE CLEANUP ABEND abend_code PSW=psw RSN=rsn
modname=module_address
```

Figure 2-23 IMS 11 new messages

Figure 2-24 shows existing messages that have been updated.

```
DFS686W IMS jobname imsid INIT/TERM (nn) FAILURE RC=xxxxyyzz
```

Figure 2-24 IMS 11 updated messages

2.5.9 Glossary of LSQA terms

Table 2-10 lists terminology used in the LSQA storage reduction enhancements.

Table 2-10 LSQA storage-related terminology

Term	Meaning
BCB	*Basic control block* BCB is an internal IMS service that is used for managing sets of identical, fixed-length control block storage areas.
CDE	*Contents directory entry* A CDE is a z/OS-defined control block, used by z/OS to track modules that have been loaded into an address space (although a module is represented by a collection of blocks: CDE, LLE, and XTLST, for simplicity, this document simply refers to CDE when discussing the z/OS structure). IMS also builds CDEs and chains them into the z/OS CDE chains. CDEs are in 24-bit LSQA (private) storage.
EOM	*End-of-memory* EOM is state that occurs after an address space has completely terminated and has been cleaned up by z/OS. EOM is a broadcast (SSI or resource cleanup routine) that allows programs to get control and perform any final cleanup for the terminating address space. **Note:** At EOM time, the address space is gone, so EOM routines cannot depend on any storage in the private region of the address space. Usually, address spaces are given other opportunities for cleanup (end-of-task exits, for example). However, cases exist in which the only cleanup processing is EOM.

Term	Meaning
IMODULE	This term refers to an IMS internal macro service that is used to perform various storage management (get/free storage) and contents supervision (load/delete modules). The IMODULE service today builds z/OS contents supervisor blocks (CDEs and related blocks) to track both modules and storage, and is the main subject of this line item. With this line item, IMODULE has an option to track GETMAINed storage using an IMS-managed tracking structure in 64-bit storage.
IPAGE	*IMS page* An IPAGE is the raw storage allocation unit of the BCB storage manager. BCB allocates an IPAGE when it is called to get a BCB control block and no free blocks are available. An IPAGE contains one or more BCB control blocks. They are usually (but not always) 4K in length.
LSQA	*Local system queue area* LSQA is a z/OS storage area designation. LSQA storage is 24-bit authorized user private storage. It is allocated from the top of the 24-bit private address range downward, and is not subject to address space region limits. LSQA storage can only be allocated by an authorized caller.
RTM	*Recovery termination manager* RTM is a component of the z/OS operating system that facilitates processing associated with abends (either by recovering them or by terminating the address space). In the context of this line item, RTM comes into play when an IMS LSQA shortage prevents it from obtaining necessary storage to complete orderly termination of the IMS address space, leading to ABEND40D end-of-memory abends.
STE	*Storage tracking element* An STE is an IMS-defined control block for tracking selected IMODULE GETMAINed areas of storage. STEs replace CDEs. They are allocated in 64-bit storage.

2.6 KBLA

Interpreting and analyzing IMS log data can be time-consuming, complex, and error-prone.

IMS provides the Knowledge-Based Log Analysis (KBLA) utilities that simplify the interpretation and analysis of IMS log data.

2.6.1 Overview of KBLA

IMS 11 provides two enhancements to the Knowledge-based Log Analysis (KBLA) utilities:

- Data entry panel scrolling

  This feature provides support for scrolling the KBLA ISPF panels, which facilitates access to all of the data entry fields and hidden data lines, regardless of the screen size.
- Multi-volume KBLA output data set allocation

  This feature minimizes the potential risk for exceeding the space allocated for an output data set, which might result in an abend.

2.6.2 Data entry panel scrolling

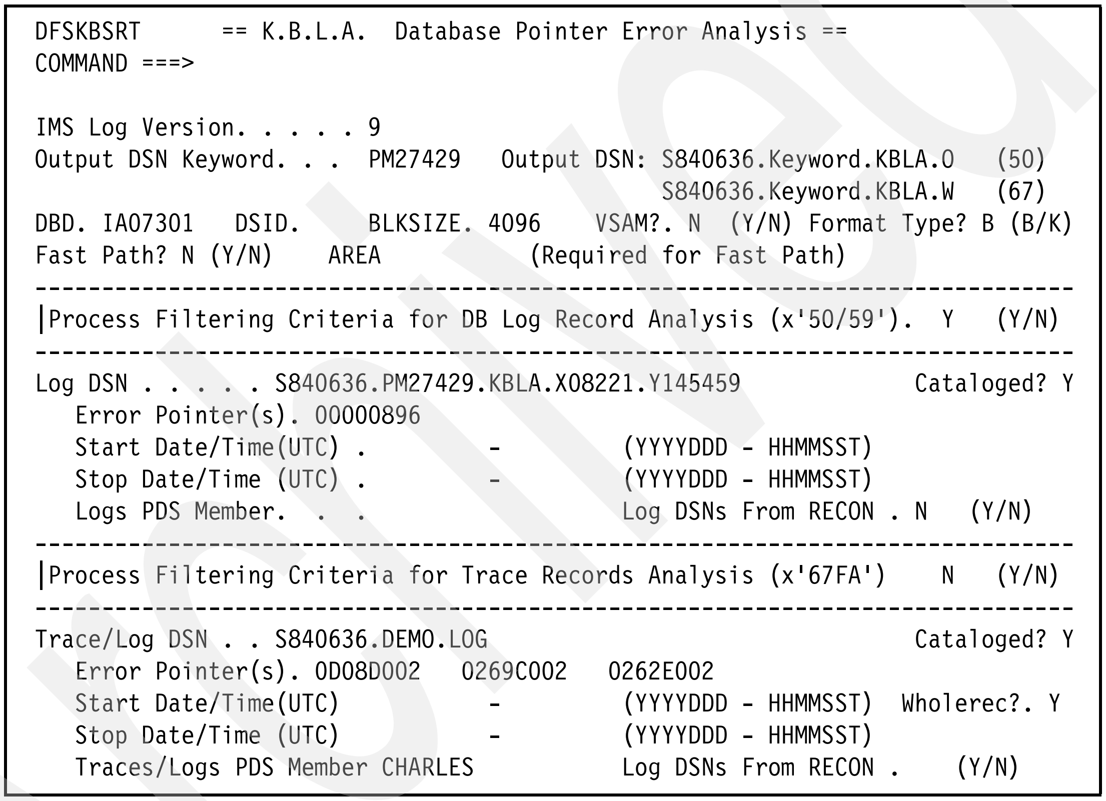
Most KBLA data entry panels do not support scrolling: If an ISPF user is using the *split-screen* function, data entry fields that are hidden by the split screen are not accessible to the user.

Panel prior to split-screen

The ISPF interface to the KBLA utilities consists of multiple ISPF data entry panels, from which terminal users enter data to be processed. Although terminal physical displays might be configurable, a typical data entry terminal displays 24 lines at a time. The 24-line limitation has several potential drawbacks:

- A logical function to be performed by an ISPF panel might require more than 24 lines to handle data.
- If the ISPF split-screen function is used, which allows for two concurrent views of data, each view might be considerably smaller than the 24-line limit.

In either case, the panel view cannot show all lines at a given time; some lines are hidden from view. Figure 2-25 shows a KBLA panel in which all data lines are visible.


```
DFSKBSRT      == K.B.L.A.  Database Pointer Error Analysis ==
COMMAND ===>

IMS Log Version. . . . . 9
Output DSN Keyword. . .  PM27429   Output DSN: S840636.Keyword.KBLA.O   (50)
                                               S840636.Keyword.KBLA.W   (67)
DBD. IA07301   DSID.     BLKSIZE. 4096    VSAM?. N  (Y/N) Format Type? B (B/K)
Fast Path? N (Y/N)    AREA          (Required for Fast Path)
------------------------------------------------------------------------------
|Process Filtering Criteria for DB Log Record Analysis (x'50/59').  Y   (Y/N)
------------------------------------------------------------------------------
Log DSN . . . . . S840636.PM27429.KBLA.X08221.Y145459            Cataloged? Y
   Error Pointer(s). 00000896
   Start Date/Time(UTC) .        -         (YYYYDDD - HHMMSST)
   Stop Date/Time (UTC) .        -         (YYYYDDD - HHMMSST)
   Logs PDS Member.  .  .                  Log DSNs From RECON . N   (Y/N)
------------------------------------------------------------------------------
|Process Filtering Criteria for Trace Records Analysis (x'67FA')    N   (Y/N)
------------------------------------------------------------------------------
Trace/Log DSN . . S840636.DEMO.LOG                               Cataloged? Y
   Error Pointer(s). 0D08D002   0269C002   0262E002
   Start Date/Time(UTC)          -         (YYYYDDD - HHMMSST)  Wholerec?. Y
   Stop Date/Time (UTC)          -         (YYYYDDD - HHMMSST)
   Traces/Logs PDS Member CHARLES          Log DSNs From RECON .    (Y/N)
```
Figure 2-25 Panel prior to split-screen

This panel can represent either:

- A panel that does not support scrolling
- A panel that supports scrolling but currently has no hidden data

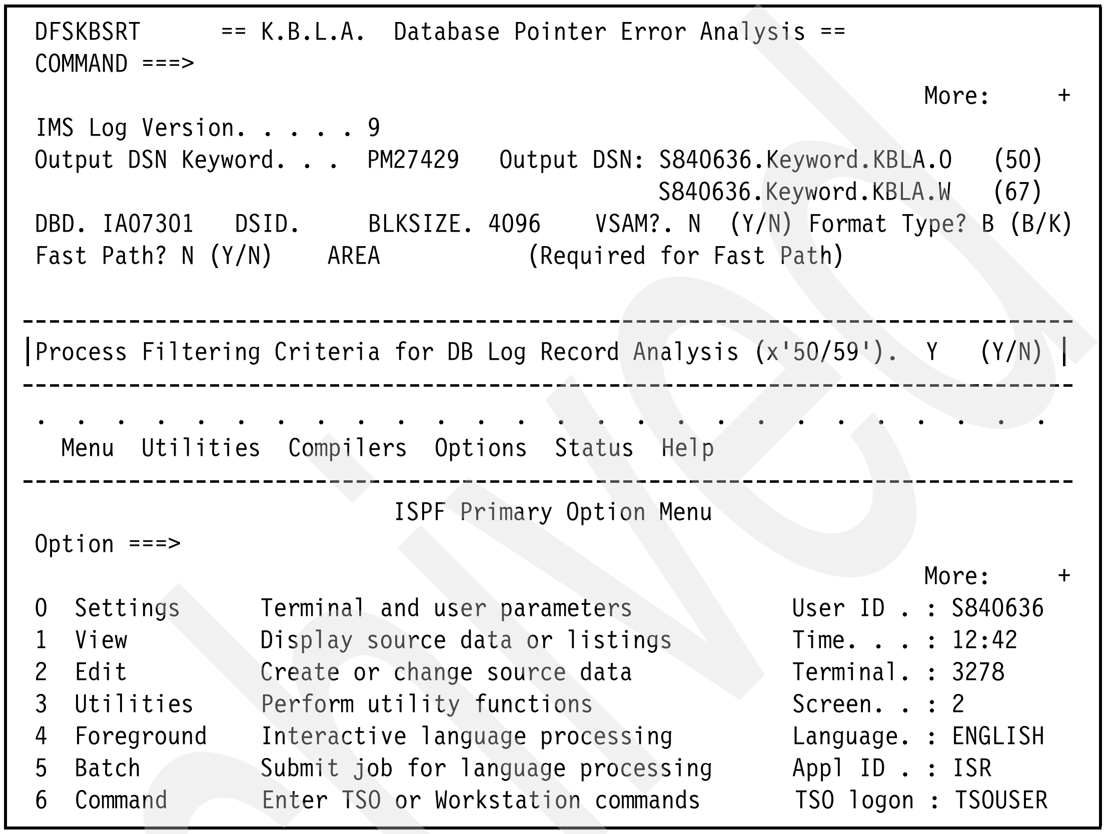
In many cases, both such panels appear at the same time.

Also, without scrolling capability, the hidden lines are not readily accessible. If lines are hidden as a result of a split-screen, a user may exit from split-screen mode to view the lines, however, scrolling provides more ease of access.

Panel with split-screen enhancement

Figure 2-26 shows a KBLA panel that supports scrolling, and is in split-screen display.

A dotted line separates the data lines belonging to the top part of the panel in from the bottom part of the panel.



```
DFSKBSRT      == K.B.L.A.  Database Pointer Error Analysis ==
COMMAND ===>
                                                                 More:     +
IMS Log Version. . . . . 9
Output DSN Keyword. . .  PM27429   Output DSN: S840636.Keyword.KBLA.O   (50)
                                               S840636.Keyword.KBLA.W   (67)
DBD. IA07301   DSID.     BLKSIZE. 4096    VSAM?. N  (Y/N) Format Type? B (B/K)
Fast Path? N (Y/N)    AREA          (Required for Fast Path)

-------------------------------------------------------------------------------
|Process Filtering Criteria for DB Log Record Analysis (x'50/59').  Y   (Y/N) |
-------------------------------------------------------------------------------
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  Menu  Utilities  Compilers  Options  Status  Help
-------------------------------------------------------------------------------
                           ISPF Primary Option Menu
Option ===>
                                                                 More:     +
0  Settings      Terminal and user parameters           User ID . : S840636
1  View          Display source data or listings        Time. . . : 12:42
2  Edit          Create or change source data           Terminal. : 3278
3  Utilities     Perform utility functions              Screen. . : 2
4  Foreground    Interactive language processing        Language. : ENGLISH
5  Batch         Submit job for language processing     Appl ID . : ISR
6  Command       Enter TSO or Workstation commands      TSO logon : TSOUSER
```

Figure 2-26 Panel with split-screen and hidden lines

The top part of the panel supports scrolling.

Note the following information about these character strings:

- `More: +`

  Indicates that hidden lines exist below the dotted separator line, and that these can be seen by pressing PF8.

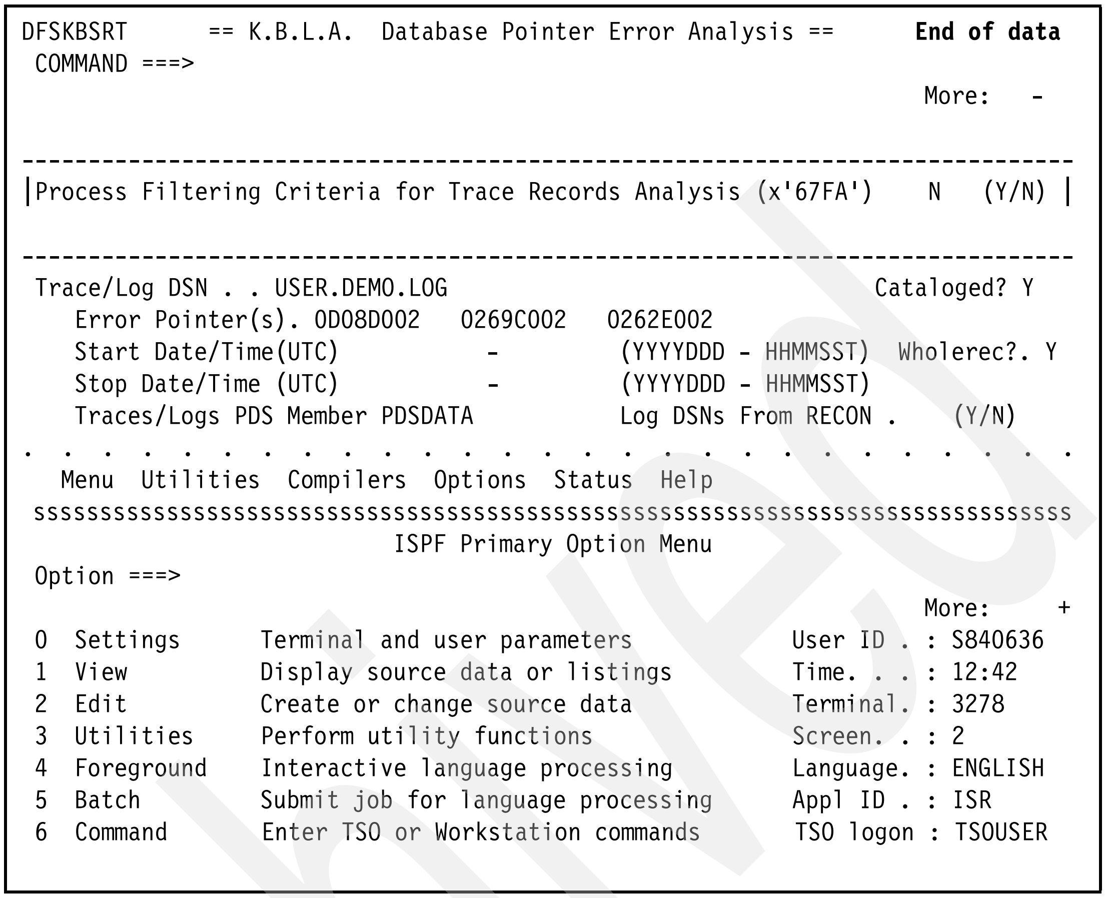
  Figure 2-27 on page 53 shows the same KBLA panel after PF8 has been pressed to show the hidden data.

- `End of Data`

  This is at the top of the panel and indicates that there are no hidden lines below the dotted separator line.

- `More: -`

  Indicates that lines are hidden, but that they are above the top line displayed in the panel. Press PF7 to see these lines.



```
DFSKBSRT      == K.B.L.A.  Database Pointer Error Analysis ==      End of data
 COMMAND ===>
                                                                    More:   -

-------------------------------------------------------------------------------
|Process Filtering Criteria for Trace Records Analysis (x'67FA')    N   (Y/N) |

-------------------------------------------------------------------------------
 Trace/Log DSN . . USER.DEMO.LOG                               Cataloged? Y
    Error Pointer(s). 0D08D002   0269C002   0262E002
    Start Date/Time(UTC)          -        (YYYYDDD - HHMMSST)  Wholerec?. Y
    Stop Date/Time (UTC)          -        (YYYYDDD - HHMMSST)
    Traces/Logs PDS Member PDSDATA         Log DSNs From RECON .    (Y/N)
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
   Menu  Utilities  Compilers  Options  Status  Help
 ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
                           ISPF Primary Option Menu
 Option ===>
                                                                    More:     +
 0  Settings      Terminal and user parameters            User ID . : S840636
 1  View          Display source data or listings         Time. . . : 12:42
 2  Edit          Create or change source data            Terminal. : 3278
 3  Utilities     Perform utility functions               Screen. . : 2
 4  Foreground    Interactive language processing         Language. : ENGLISH
 5  Batch         Submit job for language processing      Appl ID . : ISR
 6  Command       Enter TSO or Workstation commands       TSO logon : TSOUSER
```

Figure 2-27 Panel with split-screen without hidden lines

2.6.3 Multi-volume KBLA output data set allocation

Processing of large logs or multiple logs concurrently sometimes result in out-of space failures, such as SB37 abends in output data sets, because its specification was too small.

Allowing the output data sets to span multiple volumes reduces the potential for such data set out-of space type abends during KBLA-driven log analysis, and the associated rework to revise the allocations and restart the analysis jobs.

Description

Most KBLA ISPF panels generate JCL to create data sets as a part of log analysis. These data sets can contain summary reports, detail reports, and copies of log records.

KBLA can process a virtually unlimited number of input data sets concurrently and, depending on the function that the user selected, can produce a large amount of output.

KBLA has always provided an environment setup panel, which allows you to specify the amount of space to be used to allocate data sets. However, the panels did not include the ability to allocate the data sets spanning multiple volumes.

Because multiple input data sets could be concatenated, and there was no volume limitation for each of these data sets, sometimes the corresponding output generated by KBLA would not fit on a single volume, and log analysis jobs failed with size-related abends, such as SB37.

Multi-volume output data set allocation

KBLA ISPF panels have always allowed you to modify the generated JCL, so you could allocate across multiple volumes. However, you would have to make this change manually for each JCL generation.

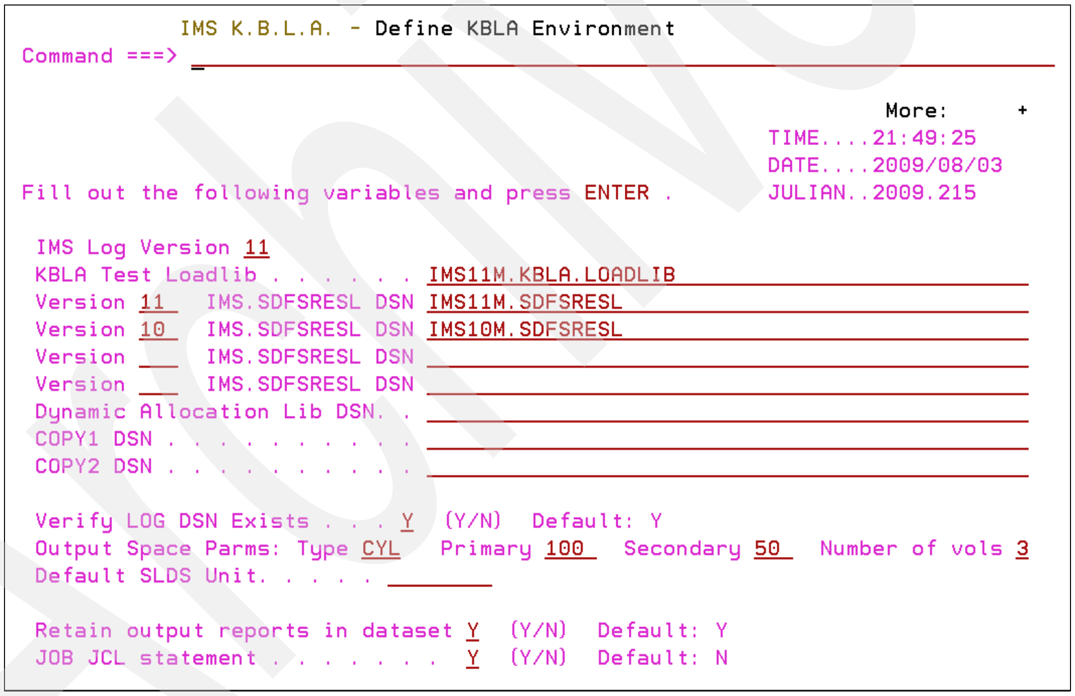
An advantageous approach (and an enhancement feature) is to allow the specification of multiple volumes on the environment setup panel, so that space allocation has to be performed only once and the specification will always be picked up in the generated JCL.

This enhancement modifies the Define KBLA Environment panel and JCL to enable the user to allocate data sets that span multiple volumes.

If a particular output data set could fit on a single volume, although the JCL specification allows spanning, such spanning would not actually occur.

Currently, although most KBLA utilities do not provide a field in which to specify the number of volumes, several of the newer KBLA panels do support this parameter. This enhancement adds universal support for the KBLA panels that did not previously support this parameter.

Figure 2-28 shows the IMS 11 KBLA environment panel that includes a field in which the number of volumes can be specified.



```
             IMS K.B.L.A. - Define KBLA Environment
Command ===> _

                                                                  More:     +
                                                           TIME....21:49:25
                                                           DATE....2009/08/03
Fill out the following variables and press ENTER .         JULIAN..2009.215

 IMS Log Version 11
 KBLA Test Loadlib . . . . . . . IMS11M.KBLA.LOADLIB
 Version 11    IMS.SDFSRESL DSN IMS11M.SDFSRESL
 Version 10    IMS.SDFSRESL DSN IMS10M.SDFSRESL
 Version ___   IMS.SDFSRESL DSN
 Version ___   IMS.SDFSRESL DSN
 Dynamic Allocation Lib DSN. .
 COPY1 DSN . . . . . . . . . .
 COPY2 DSN . . . . . . . . . .

 Verify LOG DSN Exists . . . . Y  (Y/N)  Default: Y
 Output Space Parms: Type CYL   Primary 100    Secondary 50    Number of vols 3
 Default SLDS Unit. . . . .

 Retain output reports in dataset Y  (Y/N)  Default: Y
 JOB JCL statement . . . . . . .  Y  (Y/N)  Default: N
```

Figure 2-28 Define KBLA Environment panel with field to number of volumes

Figure 2-29 on page 55 shows the DD statement generated for an output data set. This JCL includes the VOLUME= parameter, in which the number of volumes allowed for the data set is specified. If this parameter is omitted, it defaults to 1.

```
//SYSUT4    DD DSN=IMS11M.KBLA.LOG.MULTIVOL,
//  UNIT=SYSDA,
//  VOLUME=(,,,3),
//  SPACE=(CYL,(100,50),RLSE),
//  DCB=(RECFM=VB,LRECL=32756,BLKSIZE=32760),
//  DISP=(NEW,CATLG,CATLG)
```

Figure 2-29 JCL generated for multi-volume output data set

Operational characteristics

The VOLUMES= parameter is only included in the JCL for the allocation of data sets for which the values for Unit type, Primary, and Secondary space allocation are taken from the panel that is named Define KBLA Environment.

Typically, these data sets can become quite large, based on the size or number of log data sets that are used as input.

Other data sets, such as summary reports, for which the size does not fluctuate much with the size or number of log data sets, have fixed-size allocations and do not acquire these parameters from the Define KBLA Environment panel.

2.6.4 Starting the KBLA

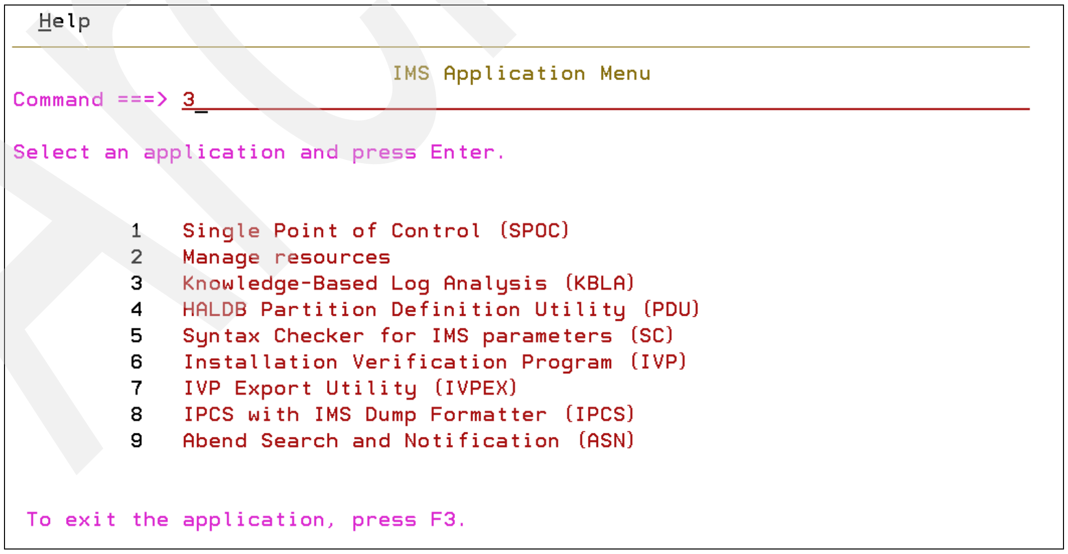
You can start the KBLA by using either of the following methods:

- Access the main KBLA panel directly by typing the following command in ISPF option 6, where HLQ is the High Level Qualifier of the IMS data sets.

  `EXEC 'hlq.SDFSEXEC(DFSKBSRT)' 'HLQ(hlq)'`

- From the IMS Application Menu (shown in Figure 2-30), access the IMS Application Menu by typing the following command in ISPF option 6:

  `EXEC 'hlq.SDFSEXEC(DFSAPPL)' 'HLQ(hlq)'`



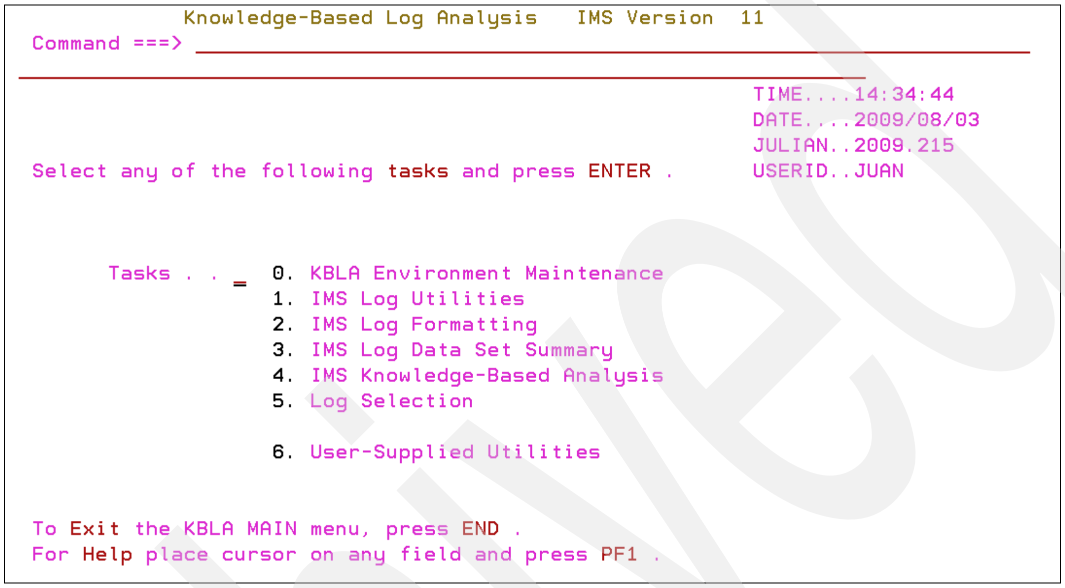
```
  Help
 ------------------------------------------------------------------------
                        IMS Application Menu
Command ===> 3_

Select an application and press Enter.


       1   Single Point of Control (SPOC)
       2   Manage resources
       3   Knowledge-Based Log Analysis (KBLA)
       4   HALDB Partition Definition Utility (PDU)
       5   Syntax Checker for IMS parameters (SC)
       6   Installation Verification Program (IVP)
       7   IVP Export Utility (IVPEX)
       8   IPCS with IMS Dump Formatter (IPCS)
       9   Abend Search and Notification (ASN)


 To exit the application, press F3.
```

Figure 2-30 IMS Application Menu

2.6.5 Using the KBLA

KBLA is a collection of IMS utilities that enable you to select, format, and analyze log records. It uses the ISPF interface to create and run the jobs for various log-related utilities, and to access other ISPF applications.

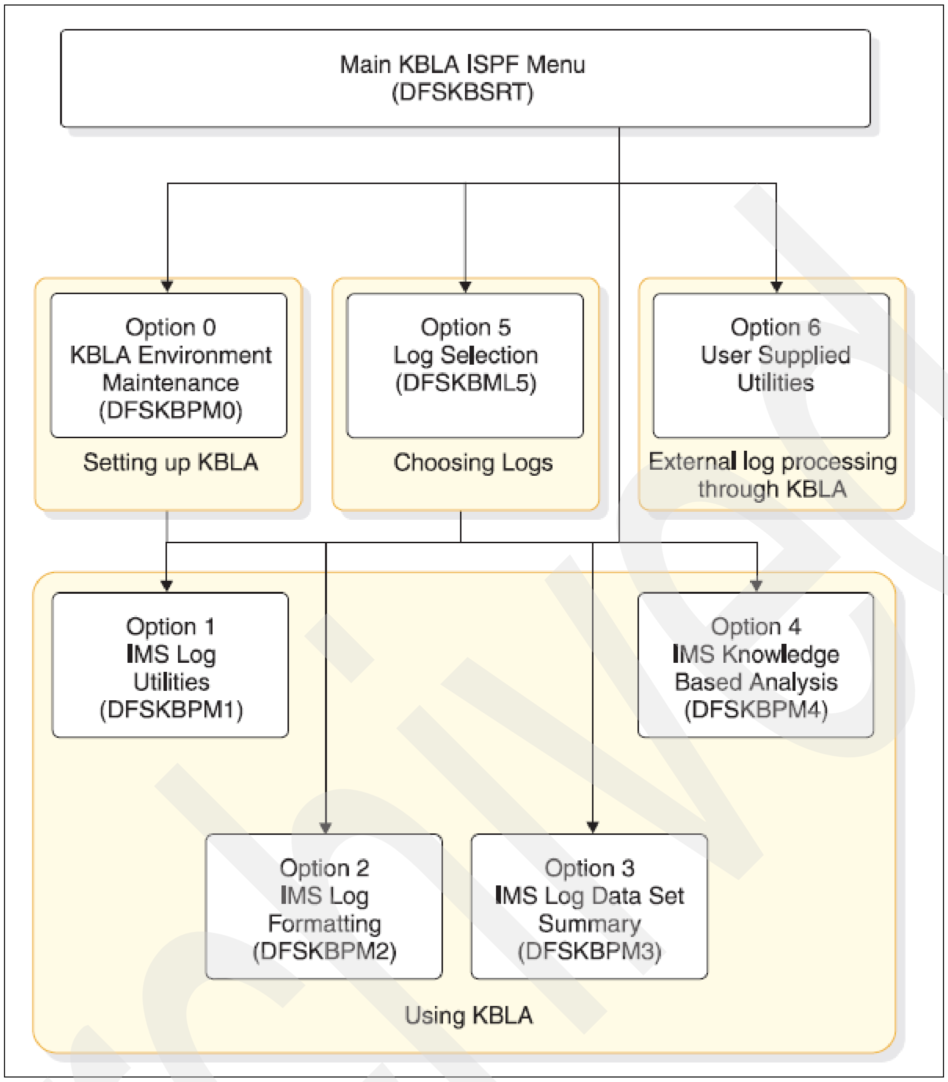
Figure 2-31 shows the KBLA Main menu panel.



```
            Knowledge-Based Log Analysis   IMS Version  11
 Command ===> _________________________________________________________________

                                                                 TIME....14:34:44
                                                                 DATE....2009/08/03
                                                                 JULIAN..2009.215
 Select any of the following tasks and press ENTER .            USERID..JUAN



       Tasks . . _  0. KBLA Environment Maintenance
                    1. IMS Log Utilities
                    2. IMS Log Formatting
                    3. IMS Log Data Set Summary
                    4. IMS Knowledge-Based Analysis
                    5. Log Selection

                    6. User-Supplied Utilities


 To Exit the KBLA MAIN menu, press END .
 For Help place cursor on any field and press PF1 .
```

Figure 2-31 KBLA Main panel

KBLA uses an ISPF panel-driven user interface to simplify JCL job creation and to prevent JCL errors. Figure 2-32 on page 57 shows the KBLA ISPF panel structure.



Main KBLA ISPF Menu (DFSKBSRT)
Option 0 KBLA Environment Maintenance (DFSKBPM0)
Setting up KBLA
Option 5 Log Selection (DFSKBML5)
Choosing Logs
Option 6 User Supplied Utilities
External log processing through KBLA
Option 1 IMS Log Utilities (DFSKBPM1)
Option 4 IMS Knowledge Based Analysis (DFSKBPM4)
Option 2 IMS Log Formatting (DFSKBPM2)
Option 3 IMS Log Data Set Summary (DFSKBPM3)
Using KBLA

Figure 2-32 KBLA ISPF panel structure

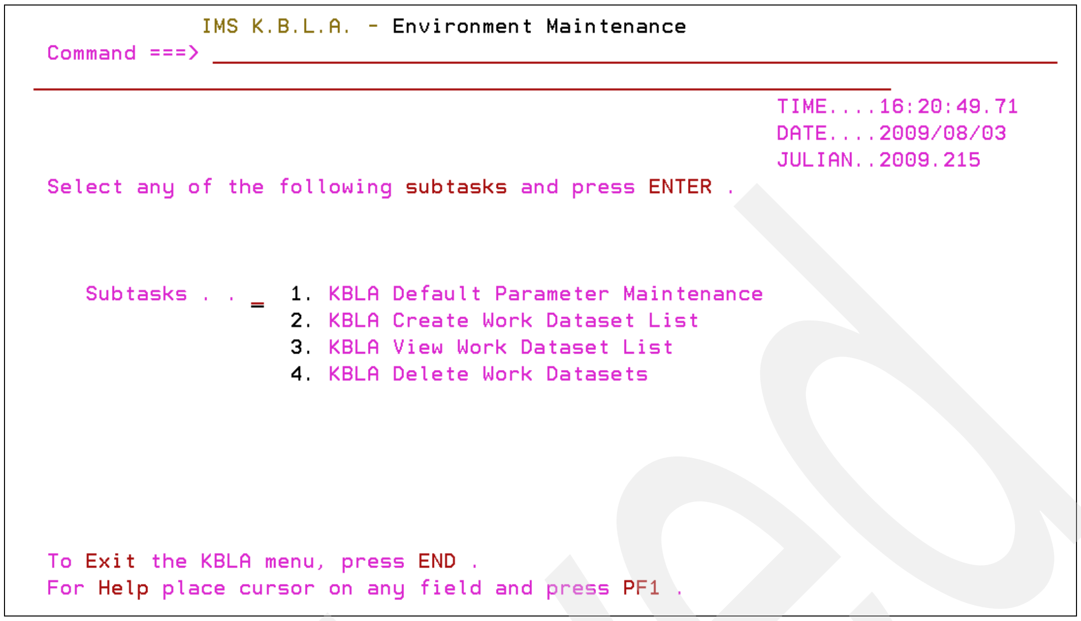
KBLA generates the JCL and control statements that are necessary to run the supported utilities. This JCL preparation enables you to focus on the output of the utility used rather than on how to code JCL to extract information.

2.6.6 KBLA environment maintenance

Figure 2-33 on page 58 shows the panel to perform global maintenance on your KBLA environment. Parameters entered in KBLA panels are stored as ISPF variables and are used by other panels, as appropriate.

Options available for KBLA environment maintenance include the following tasks:

1. Tailoring the KBLA panels according to your ISPF environment, DSN naming convention, JOB JCL requirements, IMS.SDFSRESL used, and RECON-related information
2. Creating a list of the work data sets that have been created as you use KBLA
3. Viewing the list of the work data sets
4. Deleting work data sets that are no longer needed.



```
              IMS K.B.L.A. - Environment Maintenance
 Command ===> ____________________________________________________________

                                                            TIME....16:20:49.71
                                                            DATE....2009/08/03
                                                            JULIAN..2009.215
 Select any of the following subtasks and press ENTER .


    Subtasks . . _   1. KBLA Default Parameter Maintenance
                     2. KBLA Create Work Dataset List
                     3. KBLA View Work Dataset List
                     4. KBLA Delete Work Datasets






 To Exit the KBLA menu, press END .
 For Help place cursor on any field and press PF1 .
```

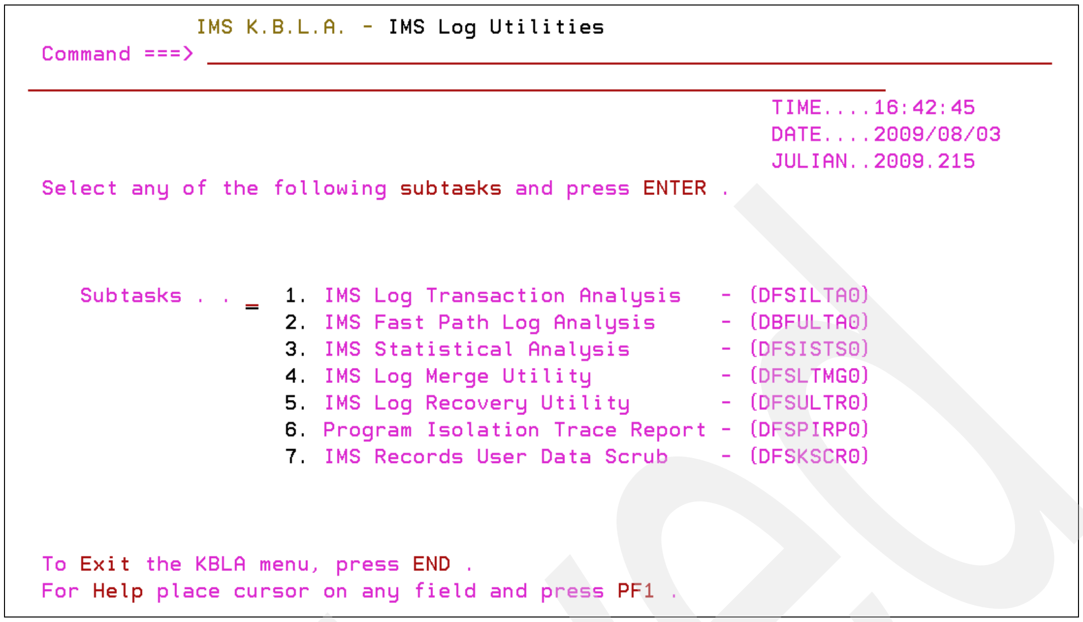
Figure 2-33 KBLA Environment Maintenance panel

Completing the KBLA Maintenance Environment panel first can be useful in order to tailor KBLA to specific requirements. If entries are not specified, KBLA provides certain fields with default values.

2.6.7 IMS Log Utilities

Figure 2-34 on page 59 shows the panel where you access the following log analysis utilities (subtasks):

1. Log Transaction Analysis utility (DFSILTA0)
2. Fast Path Log Analysis utility (DBFULTA0)
3. Statistical Analysis utility (DFSISTS0)
4. Log Merge utility (DFSLTMG0)
5. Log Recovery utility (DFSULTR0)
6. Program isolation trace record format and print module (DFSERA40)
7. IMS Records User Data Scrub utility (DFSKSCR0)

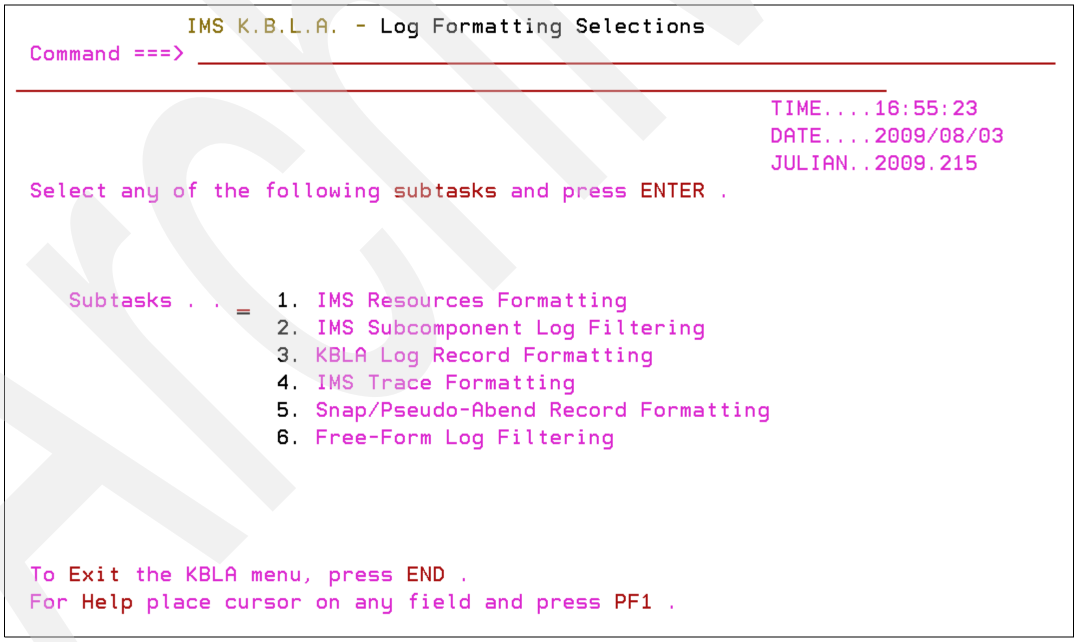


```
              IMS K.B.L.A.  - IMS Log Utilities
 Command ===> __________________________________________________________________
____________________________________________________________________
                                                            TIME....16:42:45
                                                            DATE....2009/08/03
                                                            JULIAN..2009.215
 Select any of the following subtasks and press ENTER .



    Subtasks . . _   1. IMS Log Transaction Analysis   - (DFSILTA0)
                     2. IMS Fast Path Log Analysis     - (DBFULTA0)
                     3. IMS Statistical Analysis       - (DFSISTS0)
                     4. IMS Log Merge Utility          - (DFSLTMG0)
                     5. IMS Log Recovery Utility       - (DFSULTR0)
                     6. Program Isolation Trace Report - (DFSPIRP0)
                     7. IMS Records User Data Scrub    - (DFSKSCR0)



 To Exit the KBLA menu, press END .
 For Help place cursor on any field and press PF1 .
```

Figure 2-34 IMS Log Utilities panel

2.6.8 IMS Log Formatting

Figure 2-35 shows the panel you use to extract or format log records.



```
              IMS K.B.L.A.  - Log Formatting Selections
 Command ===> __________________________________________________________________
____________________________________________________________________
                                                            TIME....16:55:23
                                                            DATE....2009/08/03
                                                            JULIAN..2009.215
 Select any of the following subtasks and press ENTER .



    Subtasks . . _   1. IMS Resources Formatting
                     2. IMS Subcomponent Log Filtering
                     3. KBLA Log Record Formatting
                     4. IMS Trace Formatting
                     5. Snap/Pseudo-Abend Record Formatting
                     6. Free-Form Log Filtering



 To Exit the KBLA menu, press END .
 For Help place cursor on any field and press PF1 .
```

Figure 2-35 IMS Log Formatting panel

Numbered options include:

1. IMS Resources Formatting
2. IMS Subcomponent Log Filtering

   A range selection panel can be invoked from which log records can be filtered by the following ranges:

   – Time stamp
   – Log sequence number (LSN)
   – Recovery tokens
   – /LOG command strings

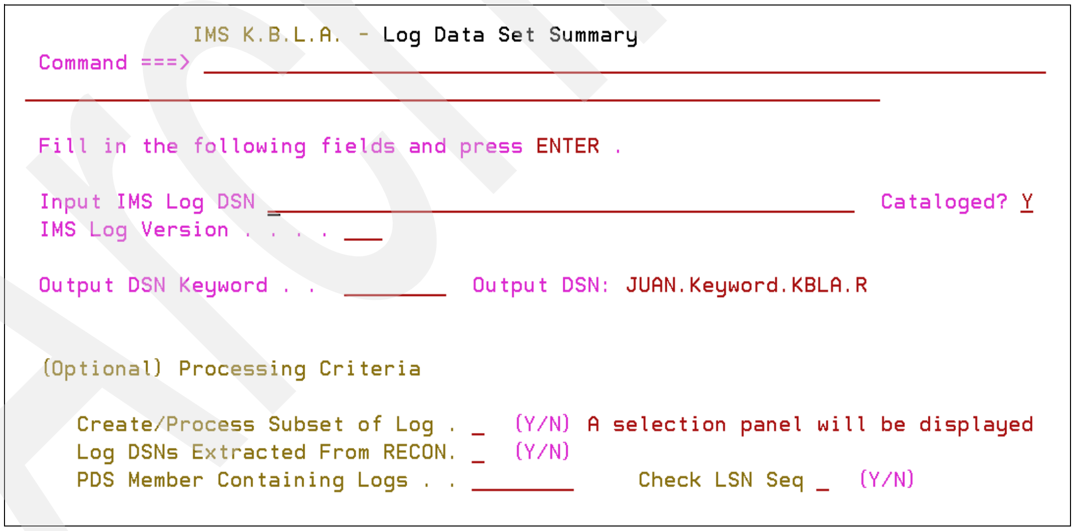
   Also a number of records to skip and process can also be specified.
3. KBLA Log Records Formatting
4. IMS Trace Formatting
5. Snap/Pseudo-Abend Record Formatting

   These include:

   – Deadlock Trace Record Analysis utility (DFSKTDL0) for K-formatted database deadlock traces
   – DFSERA10 using the Record format and print module (DFSERA30) for all other traces
6. Free-Form Log Filtering

2.6.9 IMS Log Data Set Summary

Figure 2-36 shows the option that produces a summary of the log data set that you specified, along with statistical information.



```
             IMS K.B.L.A. - Log Data Set Summary
Command ===> ________________________________________________________________
___________________________________________________________________

Fill in the following fields and press ENTER .

Input IMS Log DSN _____________________________________________  Cataloged? Y
IMS Log Version . . . . . ___

Output DSN Keyword . .  ________  Output DSN: JUAN.Keyword.KBLA.R


(Optional) Processing Criteria

   Create/Process Subset of Log . _  (Y/N) A selection panel will be displayed
   Log DSNs Extracted From RECON. _  (Y/N)
   PDS Member Containing Logs . . ________     Check LSN Seq _  (Y/N)
```

Figure 2-36 IMS Log Data Set Summary panel

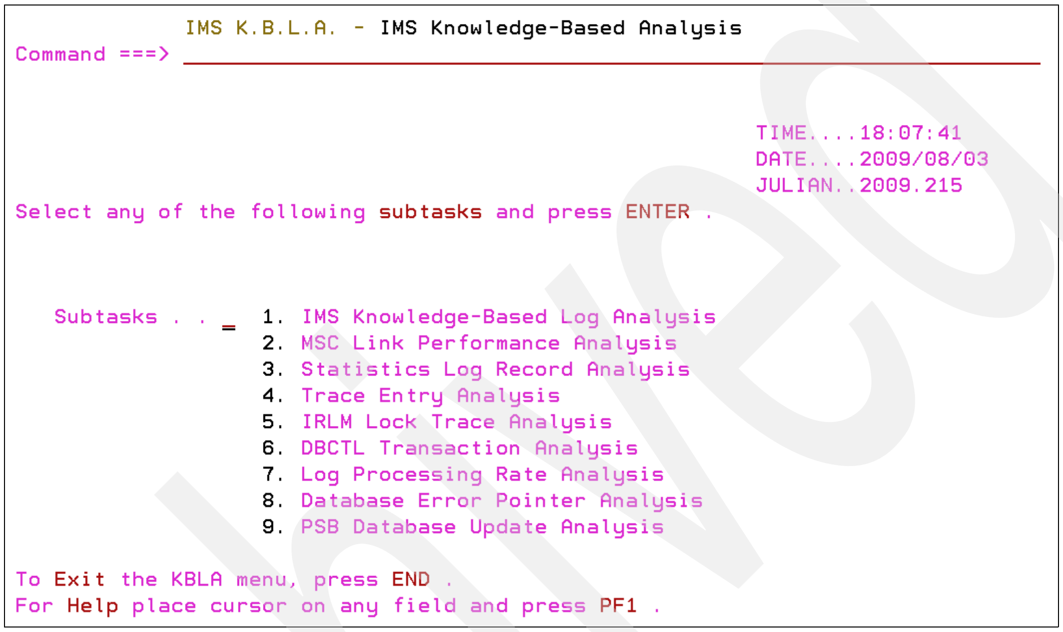
The summary function includes:

- First and last line sequence number (LSN) in the log
- Time stamp (UTC) of the first and last log record
- Total number of log records in the log data set
- Presence of internal trace records, system restarts, dump log records, system checkpoint

- Number of log records present for each record ID
- System configuration
- Transaction, program and data bases records instances

2.6.10 IMS Knowledge-Based Analysis

The option shown in Figure 2-37 provides methods for knowledge-based formatting, analysis, and interpretation of specific IMS log records.



```
                 IMS K.B.L.A. - IMS Knowledge-Based Analysis
Command ===> ___________________________________________________________________



                                                                        TIME....18:07:41
                                                                        DATE....2009/08/03
                                                                        JULIAN..2009.215
Select any of the following subtasks and press ENTER .



    Subtasks . . _   1. IMS Knowledge-Based Log Analysis
                     2. MSC Link Performance Analysis
                     3. Statistics Log Record Analysis
                     4. Trace Entry Analysis
                     5. IRLM Lock Trace Analysis
                     6. DBCTL Transaction Analysis
                     7. Log Processing Rate Analysis
                     8. Database Error Pointer Analysis
                     9. PSB Database Update Analysis

To Exit the KBLA menu, press END .
For Help place cursor on any field and press PF1 .
```

Figure 2-37 IMS Knowledge-Based Analysis panel

You may access utilities with the following options, each one with its own ISPF panel interface:

1. Log Summary utility (DFSKSUM0)
2. MSC Link Performance Formatting utility (DFSKMSC0)
3. Statistic Log Record Analysis utility (DFSKDVS0)
4. Trace Record Extract utility (DFSKXTR0)
5. IRLM Lock Trace Analysis utilities (DFSKLTA0, DFSKLTB0, DFSKLTC0)
6. DBCTL Transaction Analysis utility (DFSKDBC0)
7. Log Record Processing Rate Analysis utility (DFSKRSR0)
8. Database Error Pointer Analysis
9. PSB Database Update Analysis. Processing consists of the following steps:
   a. Optional extraction of database update records from the log by specified time ranges, skip, or process values
   b. Optional extraction of database update records by specified PSB names or database names
   c. Sorting and grouping of database update records by the following criteria:
      - PSB name
      - Recovery token

- PST number
- Database name

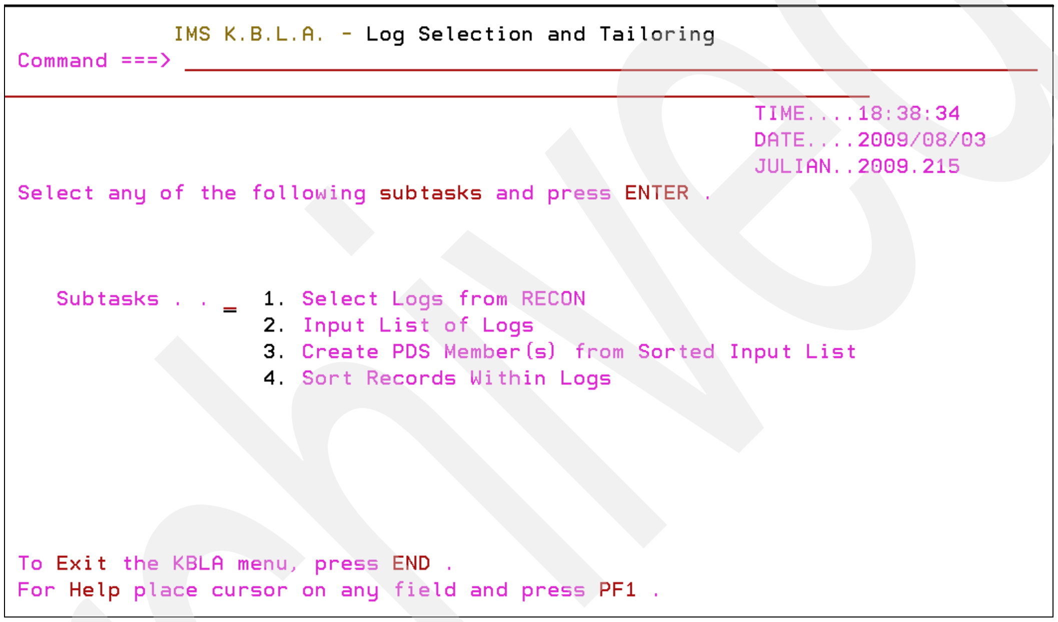
d. Counting of the database records by recovery token affinity

e. Optional sorting of the output by specified criteria

f. Optional creation of a data set that contains the sorted filtered log records

2.6.11 Log selection

Figure 2-38 shows the use of option 5, IRLM Lock Trace Analysis, to create a list of logs to be processed by KBLA. KBLA can process logs from various sources; it can also be used to sort records within log data sets.



```
             IMS K.B.L.A. - Log Selection and Tailoring
Command ===> ____________________________________________________________

                                                      TIME....18:38:34
                                                      DATE....2009/08/03
                                                      JULIAN..2009.215
Select any of the following subtasks and press ENTER .



   Subtasks . . _   1. Select Logs from RECON
                    2. Input List of Logs
                    3. Create PDS Member(s) from Sorted Input List
                    4. Sort Records Within Logs






To Exit the KBLA menu, press END .
For Help place cursor on any field and press PF1 .
```

Figure 2-38 Log selection panel

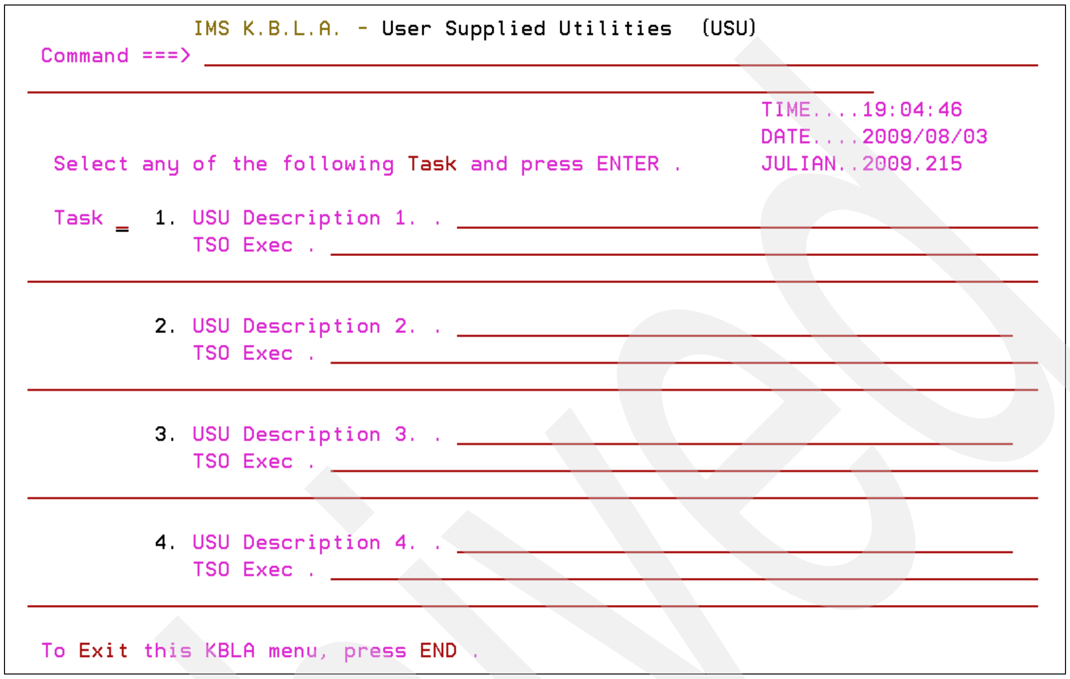
Sometimes, multiple IMS log data sets must be used. A copied or manually entered list of logs can also be used as input to KBLA. In addition to accepting a list of IMS logs, KBLA can separate the list by IMS SYSID into multiple lists, each sorted by time stamps.

The following options, listed in Figure 2-38, create and sort the final list of logs for input to KBLA:

1. Extract log data sets from the RECON data sets or as the result of a LIST.LOG command.
2. Create a member or members in your SDFSKJCL PDS that contains a list of logs to be processed by KBLA options.
3. Use the list of logs created in sub option 2 to create new sorted members in the SDFSKJCL PDS, which can be used as input by other KBLA processing options.
4. Sort records within logs by using up to four criteria, or the log sequence number (LSN), or the time stamp contained in the log suffix.

2.6.12 User Supplied Utilities

Figure 2-39 shows the use of option 6, DBCTL Transaction Analysis, to access any external log processing utility or IMS tool available in your environment.



```
              IMS K.B.L.A. - User Supplied Utilities  (USU)
 Command ===> _________________________________________________________________
______________________________________________________________________
                                                      TIME....19:04:46
                                                      DATE....2009/08/03
  Select any of the following Task and press ENTER .  JULIAN..2009.215

  Task _   1. USU Description 1. . ____________________________________
              TSO Exec . ______________________________________________
_______________________________________________________________________

           2. USU Description 2. . ____________________________________
              TSO Exec . ______________________________________________
_______________________________________________________________________

           3. USU Description 3. . ____________________________________
              TSO Exec . ______________________________________________
_______________________________________________________________________

           4. USU Description 4. . ____________________________________
              TSO Exec . ______________________________________________
_______________________________________________________________________

 To Exit this KBLA menu, press END .
```

Figure 2-39 User Supplied Utilities panel

You can launch up to four different utilities or tools from this panel by specifying the corresponding EXEC statement in the TSO Exec field.

2.7 The /DIAGNOSE command

Users should have the ability to gather diagnostic information without affecting mission-critical business operations.

For instance, gathering a console dump in a production environment is disruptive and can have a negative impact on revenue at many business sites.

Users are forced to schedule dumps during off-peak hours or are prevented from gathering this sort of information at all.

The /DIAGNOSE command takes a snapshot of IMS system resources at any time without affecting availability. The output produced can be quickly transmitted to IBM Software Support, thus avoiding the overhead of capturing and transferring a memory dump.

The /DIAGNOSE command allows users to retrieve diagnostic information for system resources such as IMS control blocks, user-defined nodes, or user-defined transactions at any time without taking a console dump.

2.7.1 /DIAGNOSE SNAP function

IMS 11 includes two new /DIAGNOSE command keywords and one new option that can help you to streamline the problem-determination process.

The /DIAGNOSE command SNAP function provides a non-intrusive alternative to creating a console dump. Using this command can decrease the time that is required to generate problem-determination data for IBM Software Support.

The /DIAGNOSE command SNAP function takes a current snapshot of system resources at any time without negatively affecting IMS. This system resource information is sent to either an online data set (OLDS) or trace data sets as type X’6701’ log records.

Because the x'6701' log record is used with no change to the mapping of the record, the existing DFSERA10 batch job can be used to extract the information. When requested, the extracted information is what should be sent to IBM Software Support.

2.7.2 SNAP BLOCK(CSCD)

The /DIAGNOSE SNAP BLOCK(CSCD) command allows the user to capture storage information for the APPC/OTMA SMQ SCD Extension control block.

This is an extension to the existing SNAP function BLOCK resource processor.

2.7.3 SNAP MODULE

The /DIAGNOSE SNAP MODULE command uses the SNAP Function MODULE Resource processor, which accepts as input a module name or list of module names, locate the module, and return the starting address and in-storage prolog information for the module.

The search process accounts for dynamically loaded modules, any module that resides in the IMS nucleus, and composite modules whose structure has been identified and internally mapped; currently, this is only the /DIAGNOSE command processing modules.

Entry point address

With this /DIAGNOSE command enhancement, users are able to determine the entry point address of the target module using the /DIAGNOSE SNAP MODULE command for the specified IMS module, and no longer have to create a console dump to set the z/OS SLIP trap command.

The entry point address can be used as input to the z/OS SLIP command when setting traps that gather information for IMS problem determination activities.

Prolog information

Another important use for the /DIAGNOSE SNAP MODULE command is to extract the prolog information for a module.

The prolog information for a module contains information about the current maintenance level of the module on the user's system. Users can identify the entry point address and capture in-storage prolog information for the specified IMS module.

The standard IMS module in-storage prolog information contains:

- Module name
- Product level
- Assembly date and time
- Last APAR ID
- Module maintenance level
- BPE version and release (for BPE based modules)
- Copyright statement

2.7.4 SNAP STRUCTURE

The /DIAGNOSE SNAP STRUCTURE command enables you to capture storage information for the DFSSQS control block storage for the specified Shared Queues structure

2.7.5 Environment

Table 2-11 lists the environments (DB/DC, DBCTL, and DCCTL) from which the /DIAGNOSE command can be issued and which keywords are valid.

Table 2-11 Valid environments for the /DIAGNOSE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/DIAGNOSE	X	X	X
ADDRESS	X	X	X
BLOCK	X	X	X
LTERM	X	-	X
MODULE	X	X	X
NODE	X	-	X
OPTION	X	X	X
SNAP	X	X	X
STRUCTURE	X	-	X
TRAN	X	-	X
USER	X	-	X

This type-1 command can be issued from an IMS terminal, a console WTOR, APPC and OTMA clients, an AOI program, MCS/EMCS consoles, and any OM command clients including SPOC.

The SNAP function of the command captures storage, both addresses and raw data, for the requested IMS control blocks, NODEs and transactions. The information in the blocks is copied to a copy storage area to avoid holding enqueues, locks, latches, and others.

The environment is further protected by a separate ESTAE routine that both protects the copy process and prevents an IMS failure.

The information is written to the OLDS. A successful completion of the command returns a DFS058I message.

2.7.6 /DIAGNOSE command syntax

Figure 2-40 shows the syntax for the /DIAGNOSE command in IMS 11.

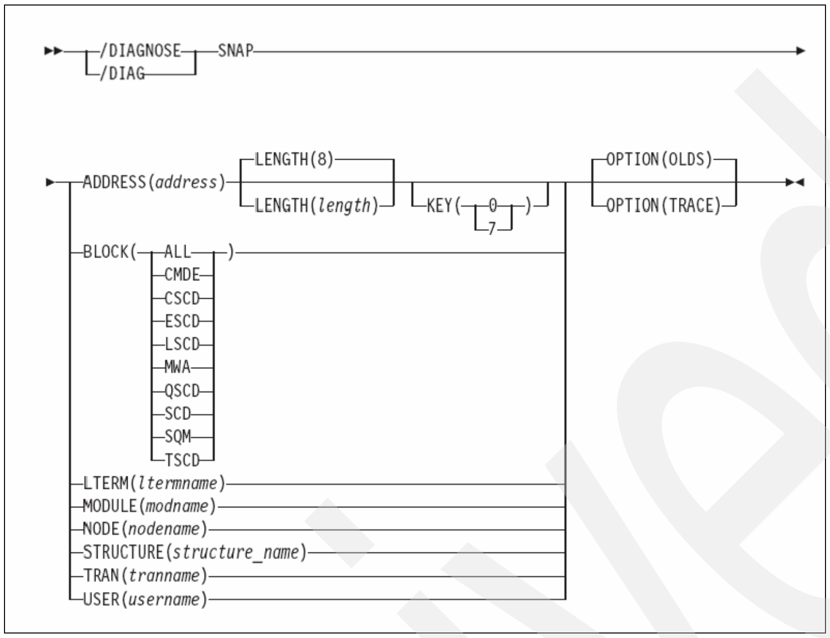


Figure 2-40 /DIAGNOSE command syntax in IMS 11

The following keywords are valid for the /DIAGNOSE SNAP command, which captures and writes storage information to the OLDS or trace data sets as type X'6701' records:

- ADDRESS

  Captures information about a specific area of storage. The address identifies the area and must specify a hexadecimal value between 0 and 7FFFFFFF. Keywords are:

LENGTH	Specifies the length of the area of storage to capture
KEY	Specifies the key of the storage to capture

- BLOCK

  Captures information for a specific IMS control block or all those available. Keywords are:

ALL	All valid control blocks currently available
CMDE	Commands SCD Extension control block
CSCD	APPC/OTMA SMQ SCD Extension control block
ESCD	Extended System Contents Directory control block
LSCD	LU 6.2 Extension to the SCD control block
MWA	Modify Work Area control block
QSCD	Queue Manager Extension to the SCD control block
SCD	System Contents Directory control block
SQM	Shared Queue Master control block
TSCD	OTMA Extension to the SCD control block

- LTERM

  Captures the same control block information as NODE but qualifications are based on the LTERM name specified.

- MODULE

  Returns the entry point address and captures in-storage prolog information about the IMS module specified.

- NODE

  Captures the following list of control block information about the node specified. Certain blocks might or might not be there depending on the type of node.

CLB	Communication Line Block
CTB	Communication Terminal Block
CTT	Communication Translate Table
CRB	Communications Restart Block
SPQB	Subpool Queue Block
CNT	Communication Name Table
CCB	Conversational Control Block
CIB	Communication Interface Block
INBUF	Input Line Buffer
OUTBF	Output Line Buffer

- OPTION

  Specifies the destination for the resource information captured by the SNAP function. The OPTION parameter is optional and has a default value of OLDS. Note:

  – If OLDS is specified, SNAP data is written to the OLDS.
  – If TRACE is specified, SNAP data is written to the trace data sets.

- STRUCTURE

  Captures information about the DFSSQS control block storage for the specified shared queues structure

- TRAN

  Captures the Scheduler Message Block (SMB) data information about the transaction specified.

- USER

  Captures the same control block information as NODE but qualifications is based on what the user specified.

2.7.7 /TRACE SET command considerations

If you do not issue the /TRACE SET ON TABLE DIAG OPTION LOG command before issuing the /DIAGNOSE command using the trace option, the /DIAGNOSE command automatically performs the following steps:

1. Turns on the DIAG trace tables.
2. Writes the output to the trace tables.
3. Turns off the DIAG trace tables.

A disadvantage of using this method is that the output from only one /DIAGNOSE command can be written to the trace data sets. Each new command overwrites the data from the last command.

To capture data from a series of /DIAGNOSE commands in a trace data set, issue the commands in the following order:

1. /TRACE SET ON TABLE DIAG OPTION LOG command
2. /DIAGNOSE commands
3. /TRACE SET OFF TABLE DIAG command

2.7.8 Examples

Example 2-7 shows several /DIAGNOSE commands to demonstrate the enhancement of the SNAP function.

Example 2-7 Examples of the /DIAGNOSE command SNAP function

```
/DIAG SNAP BLOCK(CSCD)
DFS058I 19:32:32 DIAGNOSE COMMAND COMPLETED   IMSM

/DIAG SNAP MODULE(DFSICIO0)
DFS058I 19:34:32 DIAGNOSE COMMAND COMPLETED   IMSM

/DIAG SNAP BLOCK(TEST)
DFS000I BPE0003E AN ERROR OCCURRED PARSING COMMAND DIAG SNAP
DFS000I BPE0003E AT CHARACTER 7   IMSM
DFS000I BPE0003E FAILING TEXT: "TEST)          "   IMSM
DFS000I BPE0003E UNKNOWN KEYWORD VALUE DETECTED   IMSM

/DIAG SNAP MODULE(TESTMODU)
DFS2859I DIAGNOSE COMMAND UNSUCCESSFUL - SNAP RESOURCE NOT FOUND   IMS

/DIAG SNAP STRUCTURE(TESTSTR)
DFS2859I DIAGNOSE COMMAND UNSUCCESSFUL - SQM UNAVAILABLE   IMSM
```

Example 2-8 shows the sample JCL to extract these DIAG records with DFSERA10 and print them with exit DFSERA30.

Example 2-8 JCL for printing DIAG records with exit DFSERA30

```
//JUANINIE JOB 'JUAN INIESTA',CLASS=A,MSGCLASS=J,NOTIFY=&SYSUID,
//             REGION=0M
//*****************************************************************
//P010     EXEC PGM=DFSERA10
//STEPLIB  DD DISP=SHR,DSN=IMS11M.SDFSRESL
//SYSUT1   DD DISP=SHR,DSN=IMS11M.OLP01
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
CONTROL  CNTL
OPTION   PRINT O=5,V=6701,L=2,C=M,E=DFSERA30
OPTION   PRINT O=9,V=DIAG,L=4,T=C,C=E,E=DFSERA30
END
/*
```

Example 2-9 on page 69 shows the output of this job printed with exit DFSERA30.

Example 2-9 DIAG records printed with exit DFSERA30

```
CONTROL  CNTL
OPTION   PRINT O=5,V=6701,L=2,C=M,E=DFSERA30
OPTION   PRINT O=9,V=DIAG,L=4,T=C,C=E,E=DFSERA30
END
DFSERA30  -  FORMATTED LOG PRINT                                                                             PAGE  0001
INTERNAL TRACE RECORD            ID = DIAG  SEGNO=00  RECNO = 000006ED TIME  19:32:32.453   DATE 2009.216
CSCD
 148C3000 000000   C4C6E2C3 E2C3C440  D8C90400 00000000   C4885B7E 132D4B4C  F0000000 00000000   *DFSCSCD QI......DH$=...<0.......*
 148C3020 000020   00000000 00000000  00000028 00000000   00000000 00000000  00000000 00000001   *................................*
 148C3040 000040   00000001 00000000  00000000 00000000   00000000 00000000  00000000 948C4058   *............................M. .*
 148C3060 000060   948001F8 94801B40  948C55A8 00000000   00000000 00000000  00000000 00000000   *M..8M.. M..Y....................*
 148C3080 000080   00000000 00000000  00000000 00000000   00000000 148C30C8  00000000 FFFFFFFF   *.......................H........*
 148C30A0 0000A0   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 148C30C0 0000C0   00000000                                                                      *....                            *
INTERNAL TRACE RECORD            ID = DIAG  SEGNO=00  RECNO = 000006F1 TIME  19:34:32.077   DATE 2009.216
DFSICIO0
 000484C0 000000   47F0F01C 16C4C6E2  C9C3C9D6 F0E8E8E8   60F1F161 F0F661F0  F860E800 90ECD00C   *.00..DFSICIO0YYY-11/06/08-Y.....*
DFS707I END OF FILE ON INPUT
DFS708I OPTION COMPLETE
```

2.7.9 Error conditions

Example 2-10 shows error messages that you might receive while processing the /DIAGNOSE command.

Example 2-10 Error messages received while processing /DIAGNOSE command

- Command errors

```
DFS2858E DIAGNOSE COMMAND SEVERE ERROR - [variable text, exmaples below]
DFS2857E DIAGNOSE COMMAND INTERNAL ERROR - MOD=name RSN=nnnn
```

- Block errors:

```
DFS2858E DIAGNOSE COMMAND SEVERE ERROR - BLOCK STORAGE NOT AVAILABLE (CMDE)
DFS2858E DIAGNOSE COMMAND SEVERE ERROR - BLOCK STORAGE NOT AVAILABLE (ESCD)
DFS2859I DIAGNOSE COMMAND UNSUCCESSFUL - LSCD UNAVAILABLE (LSCD)
DFS2859I DIAGNOSE COMMAND UNSUCCESSFUL - MWA UNAVAILABLE (MWA)
DFS110 COMMAND KEYWORD BLOCK (QSCD) INVALID FOR DBCTL (QSCD DBCTL)
DFS2858E DIAGNOSE COMMAND SEVERE ERROR - BLOCK STORAGE NOT AVAILABLE (QSCD NON-DBCTL)
DFS154 COMMAND BLOCK (SQM) INVALID REQUIRES COMPONENT CQS (SQM)
DFS2859I DIAGNOSE COMMAND UNSUCCESSFUL - TSCD UNAVAILABLE (TSCD)
```

- Transaction errors:

```
DFS156 TRAN KEYWORD PARAMETER IS INVALID
```

- NODE errors:

```
DFS110I COMMAND KEYWORD NODE INVALID FOR DBCTL
DFS2104 INVALID NODE NAME
```

- USER errors:

```
DFS3108I - USER KEYWORD PARAMETER IS INVALID OR MISSING
```

- LTERM errors:

```
DFS151- LTERM KEYWORD PARAMETER IS INVALID
```

- Address errors:

```
DFS2858E DIAGNOSE COMMAND SEVERE ERROR - COPY FAILED: INVALID STORAGE KEY
DFS2858E DIAGNOSE COMMAND SEVERE ERROR - COPY FAILED: STORAGE UNAVAILABLE
DFS2858E DIAGNOSE COMMAND SEVERE ERROR - COPY FAILED: DESTRUCTIVE OVERLAP
DFS2859I DIAGNOSE COMMAND UNSUCCESSFUL - INVALID ADDRESS VALUE SPECIFIED
DFS2859I DIAGNOSE COMMAND UNSUCCESSFUL - INVALID LENGTH VALUE SPECIFIED
```

2.7.10 IMSplex environment

In an IMSplex environment that uses the Resource Manager, the status recovery mode for user and node resources can be defined as local, global, or none. This definition is specified in the SRMDEF parameter of DFSDCxxx.

If specified as GLOBAL, the significant status of a resource is saved globally in the coupling facility resource structure every time the significant status changes, along with all other recoverable status for that resource.

The resource status is restored at the next logon or sign on and is available from any IMS system in the IMSplex. The resource status is copied to the local system when that resource becomes active, but is deleted from the local system when it becomes inactive.

Using the /DIAGNOSE command in this environment can therefore result in capturing the terminal control blocks from another system.

2.8 Dynamic Abend Dump Formatting exit

Before IMS 11, IMS required you to modify a z/OS module IGC0805A, CSECT IEAVADFM, in order to install its abend dump format exit module, DFSAFMD0. This module must also be linked into LPA. This updating of an operating system module makes IMS more difficult to install.

IMS 11 changes no longer use the IEAVADFM dump format exit point. Instead, IMS 11 dynamically installs a new dump format exit module DFSAFMX0 as a subsystem interface (SSI) exit that is listening for the SSI 48 “help” broadcast, which is invoked for SNAP, SYSABEND, and SYSUDUMP processing by z/OS.

With this change, DFSAFMD0 is no longer required for IMS 11 and later. But DFSAFMD0 is still necessary to provide online formatting for IMS V10 and prior versions. This enhancement is detailed in *IMS Version 11 Installation,* GC19-2438.

2.8.1 Description

In IMS 10 and earlier, you must install the IMS abend formatting module DFSAFMD0 in the host z/OS system as an IEAVADFM (Format SNAP, SYSABEND, and SYSUDUMP Dumps) exit. You are required to bind DFSAFMD0 into SYS1.LPALIB or an MLPA library. The module name DFSAFMD0 must be added to the IEAVADFM CSECT of z/OS load module IGC0805A in SYS1.LPALIB.

This process allows IMS module DFSAFMD0 to receive control during z/OS SNAP, SYSABEND, and SYSUDUMP processing so that it can format IMS control blocks in the dump. DFSAFMD0 processing is only for online dump formatting, formatting that is done at the time of the abend. It has no effect on offline dump formatting, formatting that is performed using a SDUMP or SYSMDUMP data set written at the time of the abend.

IMS 11 installs the abend formatting routine (module DFSAFMX0) dynamically. No user setup is required to install DFSAFMD0 on the host z/OS system for IMS 11 as part of the IMS installation.

Registration of the abend dump formatting exit routine with the operating system is done automatically during IMS startup.

2.8.2 Installing DFSAFMX0 module

This change is transparent to existing IMS 11 customers who already have DFSAFMD0 installed. The new SSI dump exit module coexists with the old DFSAFMD0 module.

The abend dump formatting exit routine is registered dynamically only for IMS 11 or later.

If you use earlier IMS releases, or use both IMS 11 and earlier IMS releases, you must still install the DFSAFMD0 module as part of the IMS installation if you want IMS online dump formatting.

2.8.3 DFSAFMD0 compatibility

Although DFSAFMD0 is not required for IMS 11, this module is still include to support users who point to DFSAFMD0 directly in the IMS library. DFSAFMD0 from IMS 11 can be used to provide abend formatting for IMS 10 and earlier.

You must have DFSAFMD0 installed on any z/OS LPAR where you will run previous versions of IMS 11 (either online or batch) if you want to have IMS online dump formatting support.

The version of the DFSAFMD0 module you install must be at least the same as the highest version prior to IMS 11 that you plan to run on the z/OS LPAR.

Tip: If you want to have online dump formatting available, do not uninstall DFSAFMD0 from the z/OS system until your migration to IMS 11 or later is complete, and there is no possibility that you will run an earlier release of IMS. DFSAFMD0 and the dynamic abend exit module installed by IMS 11 (DFSAFMX0) can coexist on the same system.

2.8.4 Uninstalling DFSAFMD0 module

DFSAFMD0 may be uninstalled from your z/OS system when you have completely migrated to IMS 11, however it is not required to be uninstalled.

The DFSAFMD0 module must be the highest version prior to IMS 11.

When all IMSs (control region and batch regions) are IMS 11 or later, you can remove DFSAFMD0 from SYS1.LPALIB and from the IEAVADFM CSECT of z/OS module IGC0805A.

2.8.5 Removing DFSAFMD0 from the z/OS system

After you have completely migrated to IMS 11 or later, you can remove DFSAFMD0 from the host z/OS system by performing the following steps:

1. Remove the name DFSAFMD0 from the IEAVADFM CSECT of module IGC0805A in SYS1.LPALIB.

   If you previously used the AMASPZAP utility to zap DFSAFMD0 into the IEAVADFM CSECT, you must use this utility to remove the name DFSAFMD0 from IEAVADFM.

   IEAVADFM is a table of 8-byte entries, followed by a final 4-byte entry that contains zeros to indicate the end of the exit name list.

   If DFSAFMD0 is not the last entry in the table, then in addition to removing the DFSAFMD0 entry, you must move any subsequent entries to ensure that no all-zero entries exist before the end of the table.

2. Remove DFSAFMD0 from SYS1.LPALIB or the MLPA library where DFSAFMD0 was bound.
3. Restart with CLPA to enable these changes.

2.8.6 Example to remove DFSAFMD0 from IEAVADFM

The following steps describe how to remove DFSAFMD0 from IEAVADFM:

1. Use the AMASPZAP utility to dump the current contents of IEAVADFM as shown in Example 2-11.

Example 2-11 AMASPZAP to dump contents of IEAVADFM

```
//DMPVADFM JOB ...
//STEP001  EXEC PGM=AMASPZAP
//SYSLIB   DD   DSN=SYS1.LPALIB,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
 DUMP IGC0805A IEAVADFM
```

2. Examine the contents of IEAVADFM from the AMASPZAP dump job output. Locate the entry containing DFSAFMD0 (in hex: X'C4C6E2C1C6D4C4F0') as shown in Example 2-12.

Example 2-12 Locate the entry containing DFSAFMD0

```
**CCHHR- 0022000421   RECORD LENGTH- 000BA0          +
MEMBER NAME  IGC0805A  CSECT NAME  IEAVADFM
 000000   C4C6E2C1   C6D4C4F0   D4E8C4D4   D7E7F0F0    +
          00000000   00000000   00000000   00000000
 000020   00000000   07FE0000   00000008   00000000
```

3. Use the AMASPZAP utility to replace the entry containing DFSAFMD0 with zeros.

   In the output of Example 2-12, DFSAFMD0 is the first entry in IEAVADFM, and there is one other entry following it. To remove DFSAFMD0, entry 2 must be moved to become entry 1, and entry 2 must be zapped to be all zeros, as shown in Example 2-13.

Example 2-13 Replacing DFSAFMD0 entry with zeros

```
//ZAPVADFM JOB ...
//STEP001  EXEC PGM=AMASPZAP
//SYSLIB   DD   DSN=SYS1.LPALIB,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
 NAME IGC0805A IEAVADFM
 VER  0000 C4C6E2C1C6D4C4F0
 VER  0008 D4E8C4D4D7E7F0F0
 REP  0000 D4E8C4D4D7E7F0F0
 REP  0008 0000000000000000
```

2.8.7 Changes in DFS0548E message

In IMS 11, the following explanations have been added to the DFS0548E message:

- ERROR OBTAINING DFSAFMX0 STORAGE
- ERROR ISSUING BLDL FOR DFSAFMX0
- ERROR LOADING DFSAFMX0
- DFSAFMX0 IEFSSI ADD FAILED
- DFSAFMX0 IEFSSI ACTIVATE FAILED
- DFSAFMX0 IEFSSVT CREATE FAILED
- DFSAFMX0 IEFSSVT EXCHANGE FAILED

For more information, refer to *IMS Version 11 Messages and Codes, Volume 1: DFS Messages,* GC18-9712.

3

Database enhancements

IMS 11 provides several enhancements in the Database Manager.

IMS Open Database support offers direct distributed TCP/IP access to IMS data, providing cost efficiency, enabling application growth, and improving resilience. Broadened Java tooling eases development and access of IMS data.

Database quiesce makes achieving points of consistency between databases easier, especially in an IMSplex.

IMS Fast Path Buffer Manager improves availability and overall system performance by utilizing 64-bit storage, which is also utilized for application control block (ACB) library caching and local system queue area (LSQA).

Enhanced commands and user exits simplify operations and improve availability.

In this chapter, we describe the database enhancements related to:

- IMS Open Database
- Database quiesce
- ACBLIB usability
- Database RAS
- OLR performance
- Fast Path 64-bit buffer manager
- Fast Path usability
- ODBA enhancements

3.1 IMS Open Database

IMS Open Database broadens the access to IMS databases in several ways:

- IMS Open Database eases accessibility to IMS databases from applications running on operating systems other than z/OS. WebSphere Application Server is no longer required to access IMS databases.
- IMS Open Database provides DRDA access to IMS databases.
- Improvements to ODBA access have been made, principally the ability to isolate the control region from application failures.
- A new type of database access on z/OS exists and which allows access to IMS on other LPARs in the IMSplex.

3.1.1 Overview of IMS Open Database

Before IMS 11, distributed applications had to access IMS through WebSphere Application Server and also to run WebSphere Application Server on z/OS. The connection from WebSphere Application Server on z/OS to IMS was through ODBA and this requires that you run WebSphere Application Server on the same LPAR as the IMS it connects to.

From IMS 11, it possible for distributed applications to access IMS without using WebSphere Application Server at all on any platform. It is also possible for an application running in WebSphere Application Server on z/OS to access an IMS on another z/OS image. This access is through TCP/IP and so is not restricted to the same IMSplex.

When running with WebSphere Application Server on z/OS, the application code now runs separately from the database access and so an application abend cannot lead to a u113 abend in the control region.

All existing ODBA clients can also take advantage of this u113 isolation. You do this by changing the DRA module, see "Modify DRA modules" on page 83. No application code changes are required.

Open Database components

This section discusses several of the components comprising the Open Database function.

IMS Universal drivers

IMS 11 provides a set of Java drivers that support:

- SQL calls using JDBC
- DLI calls using CCI
- SQL calls using CCI

Each of these options can use either type-2 or type-4 interfaces. The type-2 (or local) interface is through ODBA and the type-4 (or remote) interface is DRDA over TCP/IP to IMS Connect.

For more information about these drivers, refer to "Resource Adapters" on page 196.

Open Database Manager

A Common Service Layer address space, the Open Database Manager (ODBM). ODBM uses the SCI to communicate and so you must implement at least the SCI component of the IMS CSL if you have not already done so.

ODBM connects to IMS through the existing ODBA interface and so there must be one ODBM address space per LPAR, which has a control region.

IMS Connect

In IMS 11, IMS Connect acquires a major role. It is the entry point for Open Database requests that arrive through TCP/IP. These requests use the DRDA protocol.

IMS Connect uses ODBM to access the IMS databases, accessing ODBM through a new ODBM API, which operates across the IMSplex.

If you already use WebSphere Application Server on z/OS to access IMS, it can continue to fulfil this role; you do not have to configure the Open Database support of IMS Connect.

New application routes to IMS databases

IMS 11 enables new application routes to IMS data:

- From a Java application, using the new IMS Universal drivers

  Such a program can run on any platform.

- From a user-written application using TCP/IP and conforming to the DRDA protocol

  This program can run on any platform. In fact, IMS uses a subset of the DRDA protocol and has a number of structures that are specific to IMS. This means that existing DRDA applications must be changed in order to access IMS databases.

  For more about IMS use of DRDA, refer to "DRDA for DLI access" on page 196.

  A simple example DRDA application in REXX is available to download, which is described in C.1, "Sample DRDA program" on page 326.

- As an ODBM client using the ODBM API

  An ODBM client must run on z/OS within the IMSplex. This is how IMS Connect communicates with ODBM.

  This interface is available only in assembler, but it provides the ability to access an IMS on another LPAR. A simple example is available to download. It is described in C.2, "Sample ODBM program" on page 331.

Example of Open Database configurations

Many ways exist to exploit Open Database. In diagrams, we show (at a high level) several of the options you might choose.

Figure 3-1 shows a two-way data sharing IMSplex with WebSphere Application Server on z/OS.

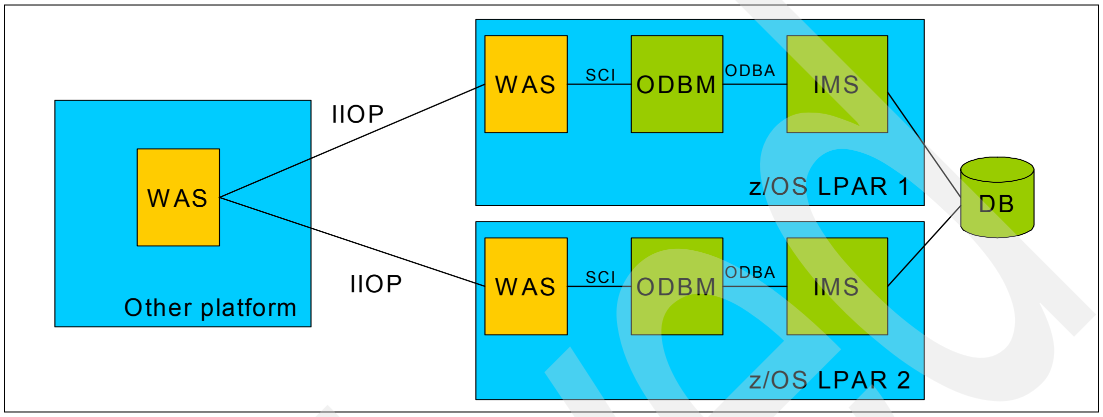


Figure 3-1 Open Database with WebSphere Application Server on z/OS

The application runs on another platform, but the IMS access is through WebSphere Application Server on z/OS. This way is similar to the WebSphere Application Server access in previous IMS versions, except that in IMS 11, application failures cannot result in u113 abends in the control region.

Although not shown in the diagram, a possibility is for WebSphere Application Server on z/OS to connect to an IMS on another LPAR. This must be through IMS Connect using DRDA over TCP/IP. You could use the type-4 interface of the IMS Universal drivers for this.

Figure 3-2 on page 79 shows the same IMSplex, but this time the access is from various distributed applications through IMS Connect. The first three applications use the IMS Universal drivers. The fourth application indicates that you may write your own program to the DRDA protocol and reach IMS data. Such a DRDA program could also run under z/OS.

In this example, each IMS Connect connects to each ODBM to provide multiple paths to the shared data.

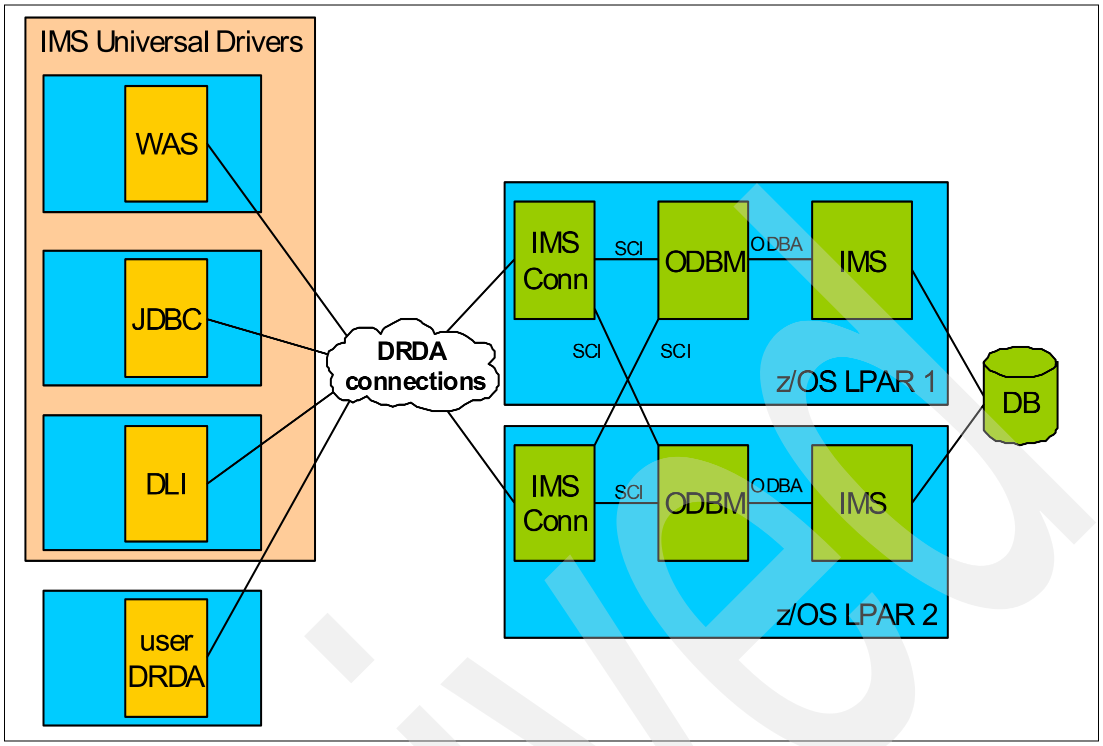


Figure 3-2 Remote Open Database access without WebSphere Application Server on z/OS

Figure 3-3 shows the same IMS configuration, but this time the accesses are from within z/OS. An ODBA application runs on LPAR 1. This application might be a business application, but it could also be a DB2 stored procedure. It can access only the ODBM on its own LPAR.

An ODBM client is running on LPAR 2. It can access either ODBM and then either IMS.

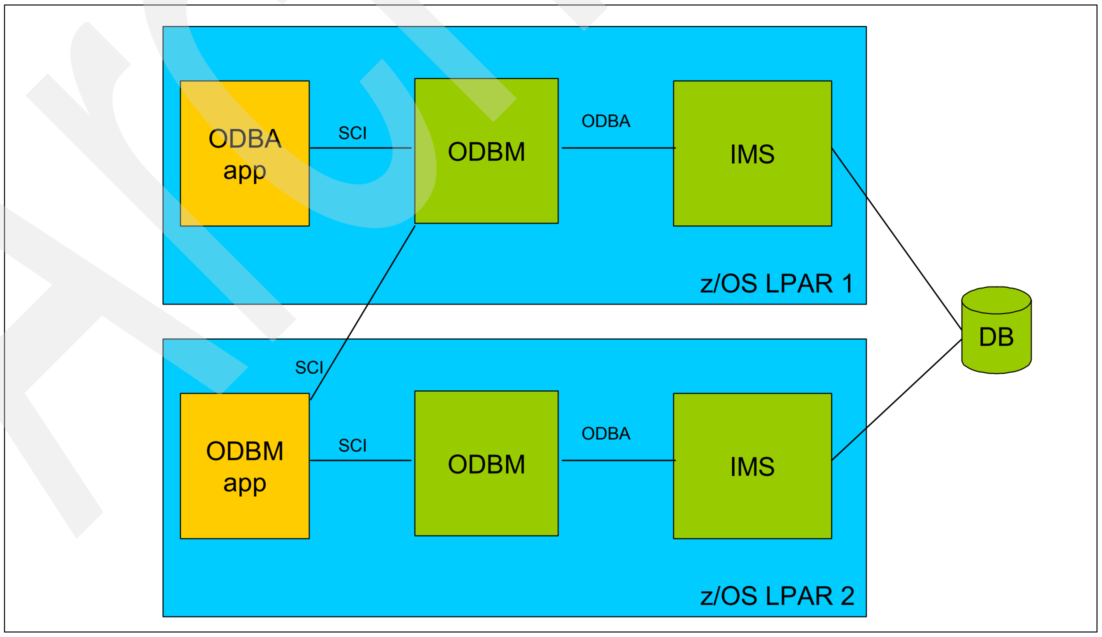


Figure 3-3 A non-Java Open Database within the IMSplex

Figure 3-4 shows how applications that are not WebSphere Application Server Java applications, running on z/OS, can connect by using IMS Open Database. At a conceptual level, the diagram is similar to Figure 3-1 on page 78. The applications connect using the type-2 interface to ODBM without the intervention of IMS Connect. Again, if one of these applications wants to contact an IMS on a different LPAR, it could do so by using the type-4 interface through IMS Connect.

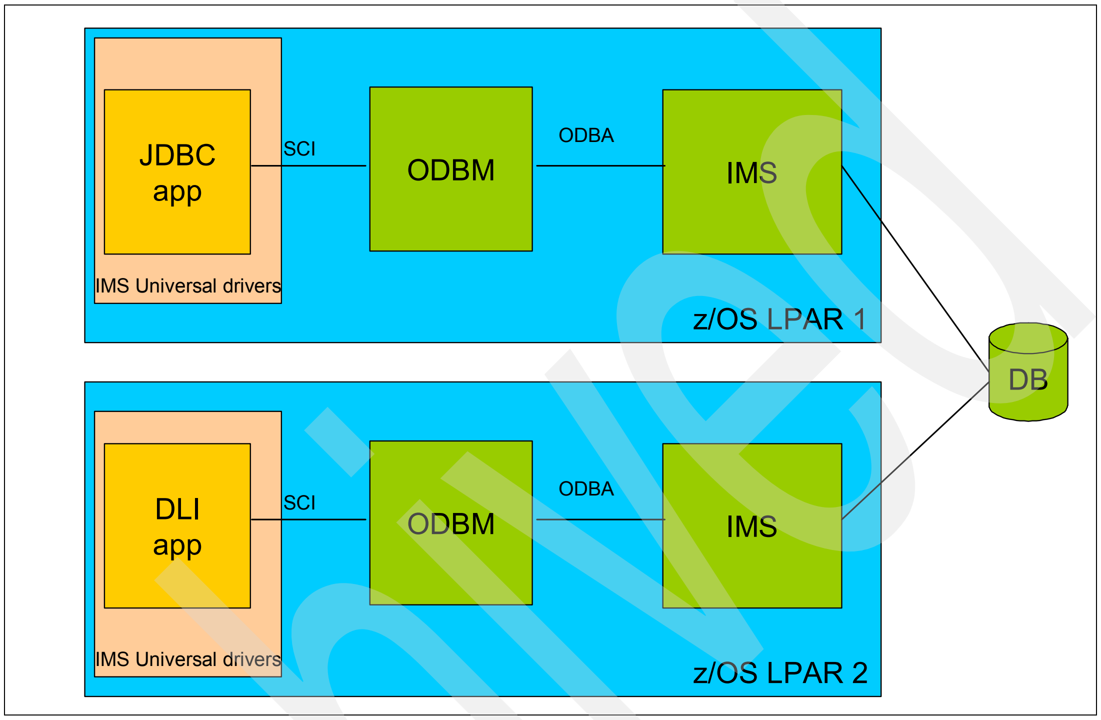


Figure 3-4 Java Open Database within the IMSplex

3.1.2 Setting up Open Database

You must perform several tasks to be able to implement Open Database:

- Set up the CSL, if you have not already done so.
- Create the ODBM task.
- Update the IMS Connect configuration, if you have already implemented IMS Connect.
- Modify DRA modules.

Set up the CSL

If you do not already have a CSL infrastructure, you must at least create SCI tasks on every LPAR that has members of the IMS data sharing group. If you plan to access IMS from LPARs that do not have a control region, you must implement SCI tasks on those LPARs also. The reason is because the SCI provides the communication between your application and ODBM.

You might also find convenience in implementing the Operations Manager, so that you can use the new type-2 IMS commands to control the ODBM address space. For examples of these commands, refer to 3.1.4, “Commands for ODBM” on page 84.

For an example of our CSL setup, refer to B.1, “IMS configuration” on page 314.

Create the ODBM task

You must create the task JCL and control statements for the ODBM tasks. You should have one ODBM task per LPAR, which has a control region. One ODBM can connect to more than one control region of the same IMSplex on the same LPAR.

Task JCL

The JCL for ODBM is short and very similar to the other CSL address spaces.

You supply two parameters:

BPECFG	The PROCLIB member containing BPE configuration statements
ODBMINIT	The suffix of PROCLIB member CSLDIxxx, which contains ODBM configuration statements

You can find sample ODBM task JCL in member CSLODBM of the SDFSISRC library.

Appendix B, “Configuration and workload” on page 313 contains the JCL we used on our system.

PROCLIB member CSLDIxxx

CSLDIxxx contains ODBM configuration statements. Most of the parameters in CSLDIxxx will be familiar to anyone who has configured a CSL address space.

You must have RRS for distributed two-phase commit or if you need ODBA access (for example from a DB2 stored procedure or from WebSphere Application Server on z/OS). RRS is used by default, so if you do not want it, you should specify RRS=N in member CSLDIxxx. Note the following settings:

- When you specify RRS=Y, ODBM connects to IMS using ODBA. In this case, both IMS Connect and the control region must also use RRS.
- When you specify RRS=N, ODBM connects to IMS using CCTL.

Parameter ODBMCFG identifies PROCLIB member CSLDCxxx which contains more ODBM configuration statements. All ODBMs in an IMSplex can share this member.

You can find sample control statements in member CSLDI000 of SDFSISRC and a good description of the parameters in *IMS Version 11 System Definition,* GC19-2444.

Also, the IMS 11 Syntax Checker understands CSLDIxxx.

Appendix B, “Configuration and workload” on page 313 contains the member we used on our system.

PROCLIB member CSLDCxxx

CSLDCxxx has two parts:

- A global section, which sets values for the IMSplex as a whole
- A local section, which defines the various ODBM address spaces and control regions

The global section describes things such as whether to retry connection to IMS, the number of threads and the number of Fast Path buffers that each thread should have.

The local section contains alias names for the various IMS control regions, known as data stores. DRDA applications must specify one of the data-store alias names to access a particular IMS. You can also override the global values for specific control regions within the local section.

You can find sample control statements in member CSLDC000 of SDFSISRC. A description can be found in *IMS Version 11 System Definition,* GC19-2444.

Again, the IMS 11 Syntax Checker can help.

Appendix B, “Configuration and workload” on page 313 contains the member we used on our system.

ODBM parameter summary

Figure 3-5 shows the relationship between the ODBM task JCL and PROCLIB members.

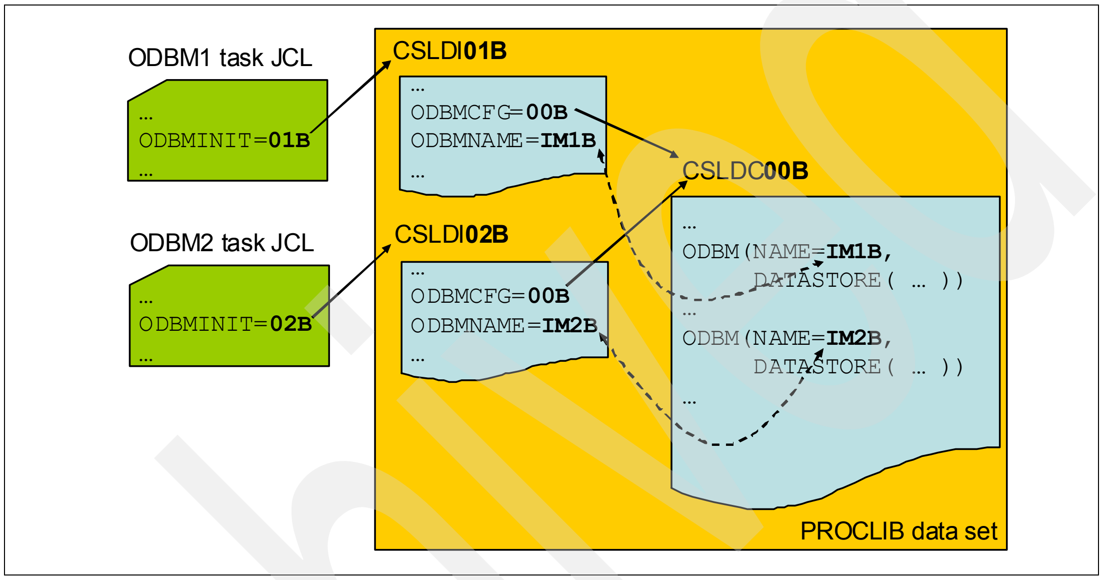


Figure 3-5 The relationship between the ODBM parameters

In this example, only two ODBMs are in the IMSplex. In the figure:

- JCL parameter `ODBMINIT=` identifies the CSLDIxxx member.
- CSLDIxxx parameter `ODBMCFG=` identifies the CSLDCxxx member, which in this case is shared.
- CSLDCxxx parameter `ODBM(NAME=` must match CSLDIxxx parameter `ODBMNAME=`.

BPE parameters for ODBM

No special BPE requirements exist for ODBM, but if you want, you may set ODBM traces and exit points.

Example B-4 on page 316 shows the parameters we used on our system.

You can find sample control statements in member BPECONFG of SDFSSLIB, and includes the ODBM traces.

For a complete description, refer to BPE configuration parameter member of the IMS PROCLIB data set, in *IMS Version 11 System Definition,* GC19-2444.

Update the IMS Connect configuration

You must use IMS Connect for IMS database access with TCP/IP, unless you use WebSphere Application Server on z/OS for this purpose.

If you have not already implemented IMS Connect, you may want to do so.

If you have implemented IMS Connect, you must add an ODACCESS control statement to your IMS Connect configuration statements if you plan to use IMS Connect with Open Database.

For more information about configuring IMS Connect, refer to:

- “Configuration member example” on page 173
- HWSCFGxx in *IMS Version 11 System Definition,* GC19-2444

Modify DRA modules

You should add the IMSPLEX= parameter to your existing DRAs and reassemble them. This information is what provides u113 isolation to ODBA applications because it causes ODBA to use ODBM instead of connecting directly to IMS. You might also want to add the ODBMNAME= parameter if you want to ensure connection to a specific ODBM.

Example 3-1 shows a DRA used on our system. We have specified only IMSPLEX because we do not have to restrict ourselves to a specific ODBM.

Example 3-1 Sample DRA for ODBA access through ODBM

```
DFSIM1B0 CSECT
DFSIM1B0 AMODE 31
         DFSPRP DSECT=NO,                                       X
               PCBLOC=31,                                       X
               DBCTLID=IM1B,                                    X
               IMSPLEX=PLEXB,                                   X
               DSNAME=IMSPSA.IMOB.SDFSRESL
         END
```

Note: Although the ODBA program connects to IMS through ODBM, connecting to an ODBM is only possible if it is running on the same LPAR. If your programs have to access an ODBM on another LPAR, you can choose between DRDA and the ODBM API.

You do not require a DRA module to access IMS Open Database through IMS Connect.

3.1.3 Security

If you are accessing ODBM through IMS Connect, you can use IMS Connect’s security features to check the user. This approach can be through SAF or through the user exit HWSAUTH0.

ODBM also has security features. If you run with RRS=Y, you have two choices: APSB security or RAS security. The same security rules apply whether you are using IMS Connect, your own ODBM client, or ODBA programs.

APSB security

If you specify DFSPBxxx parameter ODBASE=Y and use RRS, ODBM checks the user’s authority to use the PSB by checking profiles in class AIMS. This step is called APSB security, and was available for ODBA in earlier IMS versions.

RAS security

If you specify ODBASE=N, leave ODBASE to default, or run without RRS, ODBM honors the setting of the ISIS= parameter in DFSPBxxx. As in previous IMS versions, this method indicates whether SAF, the exit DFSRAS00, or no checking should be performed. This approach is known as RAS security.

If you choose SAF checking for RAS security, ODBM checks authority to use the PSB by using class IIMS. The check uses the user ID that IMS Connect (your own ODBM client) passes to ODBM.

If you are using ODBM from an ODBA application such as WebSphere Application Server or a DB2 stored procedure, the user ID will be that of the user. If you are using OBM from an ODBA application running in batch, the user ID of the job will be used.

The user ID of the ODBM address space (USER= on ODBM JCL) is used to perform security checking if the user did not pass a user ID.

3.1.4 Commands for ODBM

In IMS 11, type-2 commands available are QUERY ODBM and UPDATE ODBM, which enable you to display information about ODBM and to change the configuration, respectively.

The following sections contains examples, however for more information, refer to *IMS Version 11 Commands, Volume 2: IMS Commands N-V,* SC19-2431.

QUERY ODBM commands

The QUERY ODBM commands allow you to see how your ODBM is running:

- To check how the aliases are assigned and which are started, use:

```
QUERY ODBM TYPE(ALIAS) SHOW(ALL)
```

- To report all the information that ODBM read from PROCLIB member CSLDCxxx, use:

```
QUERY ODBM TYPE(CONFIG) SHOW(ALL)
```

- To see similar information, but sorted by data store, use:

```
QUERY ODBM TYPE(DATASTORE) SHOW(ALL)
```

- To see the current ODBM clients, use:

```
QUERY ODBM TYPE(SCIMEMBER) SHOW(ALL)
```

- To show individual ODBM threads, use:

```
QUERY ODBM TYPE(THREAD) SHOW(ALL)
```

- To show the status of any trace that is running, use:

```
QUERY ODBM TYPE(TRACE) SHOW(ALL)
```

UPDATE ODBM commands

The UPDATE ODBM commands allow you to start and stop data stores and aliases, activate traces, and change the configuration:

- To start or stop data stores or aliases use the following commands:

```
UPDATE ODBM START(CONNECTION) DATASTORE(names) | ALIAS(names)
UPDATE ODBM STOP(CONNECTION) DATASTORE(names) | ALIAS(names)
```

- To start or stop a trace use the following commands:

```
UPDATE ODBM START(TRACE) DATASTORE(names)
UPDATE ODBM STOP(TRACE) DATASTORE(names)
```

  ODBM trace records are written to memory in the same way as other CSL traces.

- To update the configuration, use the commands in the following sequence:

  ```
  UPDATE ODBM STOP(CONNECTION) DATASTORE(*)
  UPDATE ODBM TYPE(CONFIG) MEMBER(xxx)
  UPDATE ODBM START(CONNECTION) DATASTORE(*)
  ```

  This sequence reads and activates PROCLIB member CSLDCxxx. If you change the suffix xxx, you must also remember to change member CSLDIxxx or the ODBM task JCL or else ODBM will revert to the old settings when it is next started.

Notice that ODBM requires the existing data stores be stopped before the configuration can be changed.

3.2 Database quiesce

Creating a recovery point in versions prior to IMS 11 is typically done by issuing a /DBR command for the databases.

The database data sets are closed and deallocated. All programs using these databases must be terminated. These programs include BMPs and JMPs. The process creates a significant outage.

IMS 11 introduces the database quiesce function, which makes achieving points of consistency easier, especially when BMPs are running.

With IMS 11, you simply use the type-2 `UPDATE` command. There is no type-1 equivalent.

With database quiesce, the databases remain available for reading (read-access is not affected) and updates are only suspended until the quiesce point is reached. This process should typically take only a few seconds. BMPs, JBPs, and online transaction programs do not have to be terminated.

This approach simplifies the process of creating a recovery point. Database data sets are not closed or deallocated; it also shortens the process.

When you issue the command to quiesce databases, IMS suspends new update work and waits for existing update work to reach a syncpoint. When the updater has committed, IMS makes it wait until the quiesce to be released. IMS coordinates this across the IMSplex.

The two database quiesce types are:

- Quiesce and go

  IMS immediately resumes work after the quiesce point is reached. This type establishes a recovery point.

- Quiesce and hold

  IMS does not resume work until you issue a command. This type allows you to take a crisp image copy.

IMS switches the OLDS immediately before resuming work, unless you specifically ask it not to. Again, this process is coordinated across the IMSplex.

If you ask for more than one database, partition, or area to be quiesced, the quiesce does not complete until all of them are quiesced.

IMS does not wait forever for update work to reach a syncpoint. After 30 seconds, IMS abandons the quiesce. You can alter this default in either PROCLIB members: DFSCGxxx or DFSDFxxx. Example 3-2 and Example 3-3 illustrate this point.

Example 3-2 Changing the quiesce timeout in DFSDFxxx

```
<SECTION=COMMON_SERVICE_LAYER>
...
DBQUIESCETO=sss
...
```

Note: Member DFSDFxxx was introduced with IMS 10 and is identified by DFSPBxxx parameter DFSDF=.

Example 3-3 Changing the quiesce timeout in DFSCGxxx

```
...
DBQUIESCETO=sss
...
```

You can alter the time out value for an individual UPDATE command by specifying:

```
SET(TIMEOUT(xxx))
```

3.2.1 Requirements and restrictions

RECON must be at MINVERS('11.1'). This mean that once you have set RECON to MINVERS('11.1') you cannot change MINVERS to a lower level while a quiesce is in progress.

You must have a CSL with at least an SCI and an Operations Manager because database quiesce uses only type-2 commands.

You must use the CSL Resource Manager (RM) if you want to use quiesce in an IMSplex with more than one member. The RM coordinates the quiesce process. You do not have to use the Resource Manager CF structure.

A database cannot be quiesced if it is being updated by a batch job (DLI or DBB). A DLI or DBB update job that starts while one of its databases is quiesced or has a quiesce in progress that fails DBRC authorization.

A database cannot be quiesced if it requires recovery or back out.

An HALDB cannot be quiesced while it is being reorganized online and OLR cannot start while one of the partitions to be reorganized is quiesced or has a quiesce in progress.

During database quiesce, no database utilities can run, except that image copy can run while a quiesce is held.

IMS still holds the database data sets with DISP=SHR during quiesce, so your image copy job must be able to share the database data sets.

During database quiesce IMS processes no commands that can change the state of the database (for example /STA, /STO, /DBR, /DBD, /VUN, UPDATE) except the command to release a held quiesce.

If a command that can change the state of the database is already in progress when you issue a quiesce command, IMS rejects the quiesce.

All database types can be quiesced except GSAM and MSDB.

3.2.2 Commands for quiesce

IMS 11 has parameters for database quiesce on the UPDATE DB, UPDATE AREA and UPDATE DATAGRP commands.

To quiesce and go, you use START(QUIESCE), either specifying OPTION(NOHOLD) or allowing it to default.

To quiesce and hold, you use the following sequence:

```
START(QUIESCE) OPTION(HOLD)
STOP(QUIESCE)
```

For full details of these commands, refer to *IMS Version 11 Commands, Volume 2: IMS Commands N-V,* SC19-2431.

The following command sequence illustrates how quiesce and hold works.

1. From a SPOC, enter:

   ```
   UPDATE DB NAME(DI21PART) START(QUIESCE) OPTION(HOLD)
   ```

   The response is:

   ```
   DBName   MbrName    CC
   DI21PART IM1B        0
   DI21PART IM2B        0
   ```

2. From a SPOC, enter:

   ```
   QUERY DB NAME(DI21PART) SHOW(ALL)
   ```

   The response is:

   ```
   DBName   MbrName    CC TYPE     LAcc LRsdnt LclStat
   DI21PART IM1B        0 DL/I     UPD  N      ALLOCS,OPEN,QUIESCED
   DI21PART IM2B        0 DL/I     UPD  N      ALLOCS,OPEN,QUIESCED
   ```

   You can see that the database is now quiesced.

3. From an IMS terminal, enter:

   ```
   ADDPART 123456, Washer, a
   ```

   There is no response because ADDPART is an update transaction. However, transaction PART, which does not perform updates, can still run.

4. From a z/OS console, enter:

   ```
   /DIS A
   ```

   The response is:

   ```
   REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
   ...
       3 IMS11MPP  TP    ADDPART   DFSSAM04 WAIT-QSC          1
   ...
   ```

   You can see that the transaction ADDPART has started, but is waiting for the release of the quiesce.

   At this point, we could run an image copy.

5. From a z/OS console, enter:

   ```
   /RML DBRC='DB DBD(DI21PART)'
   ```

   The response is:

   ```
   LIST.DB DBD(DI21PART)
   +-----------------------------------------------------------------------------
     DB
   +  DBD=DI21PART                 IRLMID=*NULL        DMB#=17      TYPE=IMS
      SHARE LEVEL=3               GSGNAME=**NULL**     USID=0000000006
   +  AUTHORIZED USID=0000000006  RECEIVE USID=0000000006 HARD USID=0000000006
   RECEIVE NEEDED USID=0000000000
      DBRCVGRP=**NULL**
      FLAGS:                               COUNTERS:
        BACKOUT NEEDED        =OFF           RECOVERY NEEDED COUNT   =0
        READ ONLY             =OFF           IMAGE COPY NEEDED COUNT =0
        PROHIBIT AUTHORIZATION=OFF           AUTHORIZED SUBSYSTEMS   =1
        RECOVERABLE           =YES           HELD AUTHORIZATION STATE=3
                                             EEQE COUNT              =0
        TRACKING SUSPENDED    =NO            RECEIVE REQUIRED COUNT  =0
        OFR REQUIRED          =NO
        REORG INTENT          =NO
        QUIESCE IN PROGRESS   =YES
        QUIESCE HELD          =YES
      ASSOCIATED SUBSYSTEM INFORMATION:
                                 ENCODED  B/O NEEDED
        -SSID-   -ACCESS INTENT-  -STATE-   -COUNT-  -SS ROLE-
        IM1B         UPDATE          3         0      ACTIVE
   ```

   You can see that the quiesce flags in DBRC are set, but that IM1B still has the database open with update intent.

6. From a SPOC, enter:

   ```
   UPDATE DB NAME(DI21PART) STOP(QUIESCE) OPTION(NOFEOV)
   ```

   In this case we have specifically asked IMS not to switch OLDS.

   The response is:

   ```
   DBName   MbrName    CC
   DI21PART IM1B        0
   DI21PART IM2B        0
   ```

7. On the IMS terminal where the ADDPART transaction was entered, we see:

   ```
   Part Number 123456 Added To Data Base
   ```

ADDPART has now run and responded to you.

3.3 ACBLIB usability

The two enhancements to ACB library management are:

- Caching ACBs
- Dynamic allocation of ACBLIBA/B

3.3.1 Caching ACBs

IMS 11 introduces the ability to cache ACBLIB members in 64-bit storage. This does not replace the existing PSB and DMB pools, but instead acts like a buffer for ACBLIB. If an ACB must be loaded, IMS looks for it first in the cache before searching the ACBLIBs. When IMS loads an ACB from an ACBLIB, it also copies the ACB into the cache in case it is needed again. If the cache becomes full, IMS overwrites the least recently used data.

ACBLIB caching is not available in batch regions.

DEDBs and resident PSBs or DMBs are not cached because they are read only once from ACBLIB at IMS start.

Online change removes those ACBs that have changed from the cache.

ACB caching is not active, by default. You activate it by specifying the cache size (in gigabytes) by using PROCLIB member DFSDFxxx. See Example 3-4.

Example 3-4 Activating ACB caching in DFSDFxxx

```
...
<SECTION=DATABASE>
ACBIN64=1
...
```

Note: Member DFSDFxxx was introduced with IMS 10, but the database section is included with IMS 11. DFSDFxxx is identified by DFSPBxxx parameter DFSDF=xxx.

You can use the following type-2 command to monitor your ACB cache:

```
QUERY POOL TYPE(ACBIN64) SHOW(ALL)
```

Example 3-5 shows the response you might get. Among other things, the command tells you the name and size of the smallest and largest ACBs in the pool. The actual response is wider than this page.

Example 3-5 Response to QUERY POOL for ACB cache

```
PoolName Type       CC      Size     Mbrs      Used      Free Overflow     Gets
ACBIN64  Cache64     0      1024       12         0      1024        0        0
ACBIN64  Cache64     0      1024        4         0      1024        0        0

(scroll right)

Hit     Miss     Isrt      Del Lmbr        Lsize Smbr        Ssize
  0        0        3        0 IVPDB1       1024 IVPDB1I       512
  0        0        1        0 DI21PART     1024 DI21PART     1024
```

3.3.2 Dynamic allocation of ACBLIBA/B

In IMS 11, you can modify the inactive ACBLIB concatenation while IMS is running. This means you can perform the following tasks without an IMS outage:

- Compress ACB libraries.
- Change the concatenation sequence.
- Add or remove ACB libraries.

This enhancement is especially important in an IMSplex with Global Online Change because in earlier IMS versions, any of these changes would have required an IMSplex-wide outage.

You modify the inactive ACBLIB concatenation while IMS is running by defining dynamic allocation modules for the IMSACBA and IMSACBB concatenations. The macros to do this are in Example 3-6.

Example 3-6 DFSMDA macros for ACBLIBA and ACBLIBB

```
   DFSMDA TYPE=INITIAL
*
   DFSMDA TYPE=IMSACBA
   DFSMDA TYPE=DATASET,DSNAME=IMS.FIRST.ACBLIBA
   DFSMDA TYPE=DATASET,DSNAME=IMS.SECOND.ACBLIBA
*
   DFSMDA TYPE=IMSACBB
   DFSMDA TYPE=DATASET,DSNAME=IMS.FIRST.ACBLIBB
   DFSMDA TYPE=DATASET,DSNAME=IMS.SECOND.ACBLIBB
*
   DFSMDA TYPE=FINAL
END
```

Note: IMSACBA and IMSACBB must be assembled separately from other dynamic allocation modules. This is why no other DFSMDA macros appear in this example.

The main advantage of this approach is to allow changes to the offline ACB libraries. When using dynamic allocation, IMS does not allocate the offline ACB libraries, leaving you free to change the size of the library or change the concatenation if necessary.

A secondary advantage is that using dynamic allocation helps to keep the ACBLIB concatenation in DLISAS the same as in the control region.

Note: Although defining a dynamic allocation module for the staging ACB library (ACBLIB without a suffix) is already possible in IMS 10, it is only used for ACBLIB Member Online Change.

3.4 Database RAS

Three enhancements in IMS V11 improve database reliability, availability, and serviceability.

- If you issue an XRST (extended restart) call and IMS detects that a GSAM output file is empty, your application will abend u0102 with reason code C4C30001.
- Message DFS1058E provides additional information on the cause of a u845 abend.
- In cases where module DFSERA20 (SNAP call facility) is called without a user abend code or as status code, the SNAP identifies the caller in the title of the SVC dump, thereby helping to diagnose the problem. Before IMS 11, the title indicated UNK (unknown).

3.5 OLR performance

IMS 11 improves HALDB online reorganization (OLR) performance by reducing log record volume and IRLM calls. You do not have to take any action to benefit from this change in OLR logic.

OLR creates many x'50' log records for the same database block. Where possible, OLR now combines these into a single x'50' record for the entire block, reducing the amount of data that must be logged (because small x'50' records carry a significant overhead).

OLR frequently asks IRLM for locks that it already has. OLR now checks to see if a required lock is already held before calling IRLM, reducing the number of IRLM calls necessary.

Other OLR improvements include:

- Sequential access is used for *get* processing of KSDSs.
- When reorganizing a root-only database, GNP calls are no longer used because they never return a segment.
- PHIDAM index-inserts are batched at the end of each unit or reorganization so that only one busy lock (ZID)[1] is required instead of one for each index-update in the UOR.
- Block locks have been eliminated for updates to the indirect list data set (ILDS) because there can be no concurrent updaters of the ILDS.

3.6 Fast Path 64-bit buffer manager

IMS 11 introduces a Fast Path buffer manager.

If you use the Fast Path 64-bit buffer manager, you do not have to specify the DFSPBxxx parameters DBBF, DBFX, and BSIZ. Instead, the Fast Path 64-bit buffer manager autonomically allocates buffer subpools up to the maximum size specified by a parameter you provide.

In particular, the Fast Path 64-bit buffer manager allocates subpools for different DEDB CI sizes, rather than insisting on a single CI size. This approach means that you can retune your DEDBs to more efficient CI sizes without affecting other DEDBs or stopping IMS.

If you add a new Fast Path database through online change, and this database cannot fit into an existing subpool, the Fast Path 64-bit buffer manager allocates a fresh subpool of a suitable size. Previously, this would have required an IMS outage.

In fact, only DEDB read-buffers are held above the 2 GB bar; other Fast Path buffers (MSDB buffers, SDEP insert buffers, system buffers, FLD call buffers, work buffers, and buffer headers) remain in 31-bit storage.

The Fast Path 64-bit buffer manager also changes the way that IMS handles overflow buffers (OBA=). Without the Fast Path 64-bit buffer manager, access to the overflow buffers is serialized. With the Fast Path 64-bit buffer manager, parallel access is allowed. Multiple regions, threads, or both, are allowed to have their OBA buffers allocated concurrently.

3.6.1 Activating the Fast Path 64-bit buffer manager

By default, the Fast Path 64-bit buffer manager is not active. You activate it using PROCLIB member DFSDFxxx, as shown in Example 3-7 on page 92. Both parameters are required to activate the buffer manager. FPBP64M sets the maximum storage used for Fast Path buffers in 64-bit storage.

[1] The serialization process (when dealing with VSAM CA and CI splits) is called the data set busy lock (ZID) in IMS.

Example 3-7 Activating the Fast Path 64-bit buffer manager in DFSDFxxx

```
...
<SECTION=FASTPATH>
FPBP64=Y
FPBP64M=1024M
...
```

Note: Member DFSDFxxx was introduced with IMS 10, but the Fast Path section is introduced in IMS 11. DFSDFxxx is identified by DFSPBxxx parameter DFSDF=.

If you use FDBR, both the control region and FDBR must have the same setting for FPBP64.

If you retain parameters DBBF, DBFX and BSIZ in DFSPBxxx after activating the Fast Path 64-bit buffer manager, IMS ignores them. Message DFS3300I will indicate that these parameters are ignored.

3.6.2 Commands for Fast Path 64-bit buffer manager

You can use the following type-2 command to monitor your 64-bit buffers:

```
QUERY POOL TYPE(FPBP64) SHOW(ALL)
```

The response to this command is both too long and wide to fit in a page. For this reason we show only the responses from one IMSplex member and we split the response into Example 3-8, Example 3-9, and Example 3-10, by showing first the left part of the response and then the right sides.

- Example 3-8 shows the summary of pool usage.

Example 3-8 Response to QUERY POOL for Fast Path buffers, summary

```
Subpool   MbrName   CC   Size Type Tot_Buf Buf_Use Buf_Avl %Use %Ext
DBF_MAXB IM1B
DBF_TOTB IM1B                  G       480       0     480

(scroll right)

ECSA_Buf ECSA_Oth 64b_Tot 64b_Buf TimeCreate
                    1024M
    440K     209K    440K    440K
```

- Example 3-9 on page 93 shows the “DBFC” subpools which are common DEDB buffers held in 64-bit storage except for the buffer headers.

  There are two lines in this response per subpool:

  – Line A shows the totals for each subpool
  – Line B shows the details of the base section of the subpool
  – Lines E (there are none in this case) would show extensions of the subpool

  Notice that each such subpool uses a small amount of ECSA (34k) and a larger amount of 64-bit storage (depending on buffer size).

Example 3-9 Response to QUERY POOL for common Fast Path buffers

```
Subpool   MbrName   CC   Size Type Tot_Buf Buf_Use Buf_Avl %Use %Ext
DBFC0001 IM1B             512  A        80              80
DBFC0001 IM1B        0         B        80       0      80    0   50
DBFC0002 IM1B            1024  A        80              80
DBFC0002 IM1B        0         B        80       0      80    0   50
DBFC0003 IM1B            4096  A        80              80
DBFC0003 IM1B        0         B        80       0      80    0   50

(scroll right)

ECSA_Tot ECSA_Buf ECSA_Oth 64b_Tot 64b_Buf TimeCreate
     34K                     40K
                      34K             40K 2009.211 16:28:52.01
     34K                     80K
                      34K             80K 2009.211 16:28:52.01
     34K                    320K
                      34K            320K 2009.211 16:28:52.01
```

- Example 3-10 shows the DBFS subpools, which are for system use. There are multiple lines per subpool. As the example shows, these buffers are entirely in ESCA.

Example 3-10 Response to QUERY POOL for system Fast Path buffers

```
Subpool   MbrName   CC   Size Type Tot_Buf Buf_Use Buf_Avl %Use %Ext
DBFS0001 IM1B             512  A        80              80
DBFS0001 IM1B        0         B        80       0      80    0   50
DBFS0002 IM1B            1024  A        80              80
DBFS0002 IM1B        0         B        80       0      80    0   50
DBFS0003 IM1B            4096  A        80              80
DBFS0003 IM1B        0         B        80       0      80    0   50

(scroll right)

ECSA_Tot ECSA_Buf ECSA_Oth 64b_Tot 64b_Buf TimeCreate
     74K
              40K     34K                 2009.211 16:28:52.01
    114K
              80K     34K                 2009.211 16:28:52.01
    354K
             320K     34K                 2009.211 16:28:52.00
```

You can increase the maximum buffer space with the following command:

`UPDATE POOL TYPE(FPBP64) SET(LIMIT(nnnn))`

Example 3-11 shows the response you might see.

Example 3-11 Response to UPDATE POOL for Fast Path Buffers

```
Buffer Type MbrName    CC
FPBP64      IM1B        0
FPBP64      IM2B        0
```

If you use this approach, you should change member DFSDFxxx to the new value or else IMS reverts to the old value at the next cold start.

3.7 Fast Path usability

In addition to the Fast Path 64-bit buffer manager, IMS 11 introduces other Fast Path enhancements, which are discussed in this section.

3.7.1 Manually opening DEDB areas

IMS 11 extends the OPTION(OPEN) parameter of the type-2 UPDATE command to DEDBs. This extension enables you manually to open DEDB areas before an application requires them, even if the area is not registered in DBRC as PREOPEN. Previously, the OPTION(OPEN) parameter was available only for Full Function databases. The following commands are examples:

- Opens all the areas in the named DEDB:

  ```
  UPDATE DB NAME(names) AREA(*) START(ACCESS) OPTION(OPEN)
  ```

- Opens the named areas:

  ```
  UPDATE AREA NAME(names) START(ACCESS) OPTION(OPEN)
  ```

Areas opened manually by this command are not automatically reopened when IMS restarts. There is no corresponding change to the type-1 /STA command.

3.7.2 Miscellaneous enhancements

Other enhancements are to diagnostic information and a processor improvement:

- Abend U1026, subcode ‘5A’x provides more diagnostics in the EPST segment work area.
- IMS no longer issues the following messages when no MSDBs are defined:

  ```
  DFS2555I “NO SUFFIX FOR MSDB MEMBER DEFINED”
  DFS2716I “NO MSDBS FOUND-NO MSDB CHECKPOINT TAKEN”
  ```

- When rescheduling an MPP, IMS does not release and reacquire a segment work area unless it must do so (because the size has increased).
- APAR PK82285 allows the ability to turn the FPBP64 buffers statistics on or off.

3.7.3 Compatibility changes

IMS ignores the DFSPBxxx parameter UHASH. In IMS 11, the UHASH is not listed in the DFS1929I message that is written when IMS is started. There is no reason to include UHASH because the value is always DBFLHSH0 no matter what is specified in the UHASH parameter.

3.8 ODBA enhancements

IMS 11 introduces two OBDA enhancements: the u113 failure isolation and a subfunction CONNECT to the CIMS call.

We describe the u113 failure isolation provided by IMS Open Database in 3.1.2, "Setting up Open Database" on page 80 because this function requires ODBM.

The IMS 11 subfunction, CONNECT to the CIMS call, allows you to connect to more than one IMS in a single call. Previously you would have had to code several CIMS INIT calls.

Another useful feature of CIMS CONNECT is that you can specify all the parameters that are usually be found in the DRA, which means that you can connect without creating a DFSxxxx0 module. This enhancement does not require you to implement Open Database. It is retrofitted by PTF to earlier IMS versions.

More information is available in the following locations:

- For IMS 9, see Table A-3 on page 307.
- For IMS 10, see Table A-4 on page 308.

4

Transaction Manager enhancements

In this chapter, we describe the benefits of the Transaction Manager enhancements of IMS 11, including:

- Type-2 QUERY commands
- APPC enhancements
- OTMA enhancements
- OTMA type-2 commands
- Transaction expiration
- Multiple Systems Coupling enhancements
- Full function response mode recovery
- Shared queues scheduling enhancement
- Dynamic storage private buffer pool enhancement

4.1 Type-2 QUERY commands

IMS Version 11 (IMS 11) can display information about all IMS TM resources by using a type-2 QUERY command, improving resource manageability. Prior to IMS 11, you could query the attributes and status of transactions and MSC resources by using a QUERY command, but you could not use it to query other types of TM resources. Instead, you had to issue a type-1 /DISPLAY command to obtain the information.

The QUERY command enables you to query the attributes and status of all IMS TM resources, including LTERMs, NODEs, USERs, and USERIDs. You can query individual resources by listing one or more names in the command, or you can use a wildcard asterisk (*) character to reference all resources of the specified type that are known to an IMS system.

To display TM resource attributes and status, you issue the type-2 QUERY command from an OM API. This capability is helpful when you query all TM resources (versus only a subset of them as before). It is also helpful in that the attributes and status that you could see from issuing multiple type-1 /DISPLAY commands is now consolidated in one type-2 QUERY command response. This approach is more efficient.

This section discusses each of the type-2 QUERY commands. For more information, refer to *IMS Version 11 Commands, Volume 2: IMS Commands N-V,* SC19-2431.

4.1.1 List of commands

This section discusses command functionality, available resource information, filter capability, syntax, and parameters. We also compare these QUERY commands to existing equivalent type-1 commands, and provide example command input and output.

The type-2 QUERY commands that display information about TM resources in IMS 11 are:

- QUERY LTERM
- QUERY NODE
- QUERY USER
- QUERY USERID

QUERY LTERM

Use the QUERY LTERM command to display the attributes and status of both static and Extended Terminal Option (ETO) logical terminals. You can also issue this command on an XRF-alternate IMS, but only information that is local to that IMS is displayed (no global support is available when QUERY is issued on an XRF-alternate IMS). A *static terminal* is a terminal that was created by the IMS SYSGEN process; an *ETO logical terminal* is a terminal that has been created dynamically.

Prior to IMS 11, to retrieve information about LTERMs, you had to issue the type-1 /DISPLAY LTERM command. Although the output of this command showed the LTERMs that were associated with the IMS system, it gave limited resource information about them. This output was limited to LTERM type, whether it was locked, stopped, in the process of purging messages, or experiencing a queue error. In a shared queues environment, you could determine the EMHQ and global queue count information. Using the QUERY LTERM command, you can display all of this same information and additional data, including:

- Input/output component numbers
- Age of messages on the shared queue
- MSNAME associated with a remote LTERM
- Associated node name

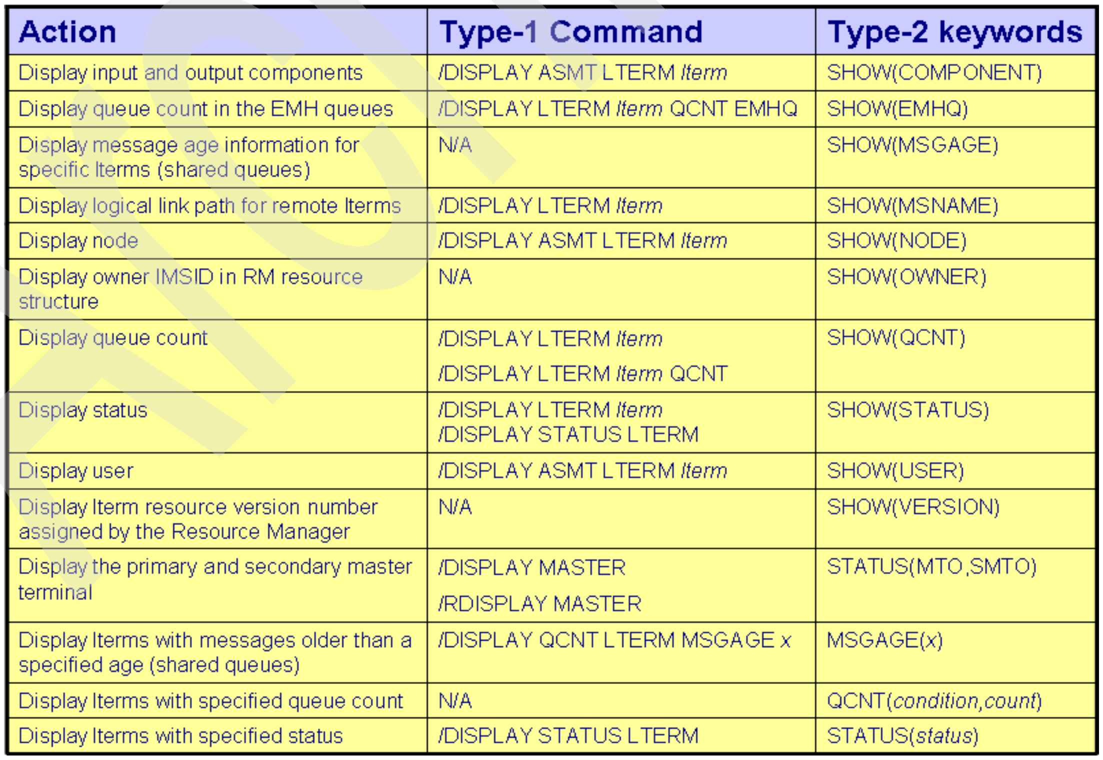
- Associated user name
- Owning IMS system
- Version number

In IMS 11, the QUERY LTERM command can be issued with various filters to show the information that you are specifically interested in without having to wade through the extra data that you would see in previous versions of IMS. When issuing this command, you can include parameters that filter the output to show only the LTERMs that have messages of a minimum specified age, or those that have queue counts with a designated qualification. For example, you might issue the QUERY LTERM command to display LTERMs that have messages queued to them and that are at least seven days old. Another example might be to issue the QUERY LTERM command to show messages that have a queue count of greater than 50 messages. As you can see, this command simplifies the output by showing only the information that interests you.

You may use the QUERY LTERM command to determine the same LTERM status information as with the type-1 /DISPLAY command, and the following LTERM status values:

- Master terminal for the local IMS
- Secondary master terminal for the local IMS
- Remote and accessible only through MSC links
- Input or output is stopped
- Exists in the resource structure managed by RM
- Associated user or node is active in an IMSplex
- Associated user or node is owned by an IMS system in an IMSplex

Prior to IMS 11, to see certain LTERM resource information you had to issue several different type-1 /DISPLAY commands; with IMS 11, you use only one QUERY LTERM command. Figure 4-1 on page 99 summarizes the resource information that is displayed with one QUERY LTERM command (in place of the several equivalent type-1 /DISPLAY commands).



Action	Type-1 Command	Type-2 keywords
Display input and output components	/DISPLAY ASMT LTERM *lterm*	SHOW(COMPONENT)
Display queue count in the EMH queues	/DISPLAY LTERM *lterm* QCNT EMHQ	SHOW(EMHQ)
Display message age information for specific lterms (shared queues)	N/A	SHOW(MSGAGE)
Display logical link path for remote lterms	/DISPLAY LTERM *lterm*	SHOW(MSNAME)
Display node	/DISPLAY ASMT LTERM *lterm*	SHOW(NODE)
Display owner IMSID in RM resource structure	N/A	SHOW(OWNER)
Display queue count	/DISPLAY LTERM *lterm* /DISPLAY LTERM *lterm* QCNT	SHOW(QCNT)
Display status	/DISPLAY LTERM *lterm* /DISPLAY STATUS LTERM	SHOW(STATUS)
Display user	/DISPLAY ASMT LTERM *lterm*	SHOW(USER)
Display lterm resource version number assigned by the Resource Manager	N/A	SHOW(VERSION)
Display the primary and secondary master terminal	/DISPLAY MASTER /RDISPLAY MASTER	STATUS(MTO,SMTO)
Display lterms with messages older than a specified age (shared queues)	/DISPLAY QCNT LTERM MSGAGE *x*	MSGAGE(*x*)
Display lterms with specified queue count	N/A	QCNT(*condition,count*)
Display lterms with specified status	/DISPLAY STATUS LTERM	STATUS(*status*)

Figure 4-1 Type-1 /DISPLAY and type-2 QUERY LTERM SHOW() equivalent commands

The syntax of the QUERY LTERM command is shown in Figure 4-2.

```
QUERY LTERM NAME(ltermname1, ltermname2,...)
MSGAGE(number)
QCNT(qualifier,number)
OPTION(MSGQ | EMHQ)
SHOW(LOCAL | GLOBAL | attribute(s))
STATUS(status)
```

Figure 4-2 QUERY LTERM command syntax

The parameters of this command are:

- NAME() is where you specify one or more LTERM names, or a wildcard asterisk (*) character.
- MSGAGE() is a filter for displaying only LTERMs that have a minimum age (0 - 365 days) on the shared queue.
- QCNT() is a filter for only displaying LTERMs with a specific queue count. Valid values for this parameter are one of these qualifiers followed by a number:
  - LT (less than)
  - LE (less than or equal to)
  - GT (greater than)
  - GE (greater than or equal to)
  - EQ (equal to)
  - NE (not equal to)
- OPTION() indicates the queue from which to draw the count. You can only validly specify this parameter when you have also specified the MSGAGE() or QCNT() parameters.
- SHOW() indicates whether to display local LTERMs, global LTERMs or specific attribute values of the LTERMs.
- STATUS() is used as a filter to display LTERMs that have a specific designated status.

As you can see, output from several /DISPLAY commands has been consolidated into output from a single QUERY LTERM command. We now show how the output is consolidated with the following four examples.

- Component, node and user information associated with LTERM resources
- LTERM queue counts and types
- Primary and secondary master LTERMs
- Consolidated LTERM resource information

Component, node and user information associated with LTERM resources

Let us say you have to determine the input/output component numbers, node names, and user names associated with all LTERMs defined to an IMS system. Prior to IMS 11, you would have had to issue the type-1 /DISPLAY ASMT command to obtain this information, as you can see in Example 4-1 on page 101. Looking at the output, you can see the columns labeled IN-TERMINAL and OUT-TERMINAL list the component numbers with the node information; the column labeled USER is available for displaying associated user names.

Example 4-1   Output of /DISPLAY ASMT command

```
*988 DFS996I *IMS READY*
 R 988,/DIS ASMT LTERM ALL
 DFS000I     LTERM    IN-TERMINAL  OUT-TERMINAL  USER
 DFS000I     WTOR         1-  SC       1-  SC
 DFS000I     USER2    USER2   -1   USER2   -1
 DFS000I     PMASTER  PMASTER1-1   PMASTER1-1
 DFS000I     DFSTCFI      4-   1   PMASTER1-1
 DFS000I     HOWARD   USER1   -1   USER1   -1
 DFS000I     SMASTER      2-   1       2-   1
 DFS000I     DFSTCF       4-   1       4-   1
 DFS000I     USER1    USER1   -1   USER1   -1
 DFS000I     IVPSPL1      3-   1       3-   1
 DFS000I     IVPPRT1      2-   1       2-   1
 DFS000I     *09198/193728*
*989 DFS996I *IMS READY*
```

LTERM queue counts and types

Next, let us say you have to determine queue count information for the LTERMs known to an IMS system, and the types of LTERMs they were. You would have had to issue a /DISPLAY LTERM command, such as in Example 4-2. You can see in the command output that only a couple of LTERMs associated with the IMS system have messages on the queue, and most of the LTERMs are of a static type.

Example 4-2   /DISPLAY LTERM command output

```
*989 DFS996I *IMS READY*
 R 989,/DIS LTERM ALL
 DFS000I     LTERM     ENQCT   DEQCT    QCT
 DFS000I     WTOR         18      18      0  STATIC
 DFS000I     USER2         0       0      0  STATIC
 DFS000I     PMASTER      22       0     22  STATIC
 DFS000I     DFSTCFI       0       0      0  STATIC
 DFS000I     HOWARD        0       0      0  STATIC
 DFS000I     SMASTER      50       0     50  STATIC
 DFS000I     DFSTCF        0       0      0  STATIC
 DFS000I     USER1         0       0      0  STATIC
 DFS000I     IVPSPL1       0       0      0  STATIC
 DFS000I     IVPPRT1       0       0      0  STATIC
 DFS000I     *09198/193833*
*990 DFS996I *IMS READY*
```

Primary and secondary master LTERMs

Continuing with our scenario, you then have to determine which LTERMs were the primary and secondary master terminals. You would issue a /DISPLAY MASTER command as in Example 4-3 on page 102. As you can see, the PMASTER and SMASTER LTERMs are the primary and secondary terminals, respectively.

Example 4-3 /DISPLAY MASTER command output

```
*991 DFS996I *IMS READY*
 R 991,/DIS MASTER
 DFS000I     LTERM PMASTER
 DFS000I     PTERM NODE PMASTER1
 DFS000I     LTERM SMASTER
 DFS000I     PTERM    2-   1
 DFS000I     *09198/194048*
*992 DFS996I *IMS READY*
```

Consolidated LTERM resource information

With IMS 11, the output shown in each of the command examples can be seen with just one type-2 QUERY LTERM command. As you can see in Example 4-4, all of the LTERM information we required in our scenario are listed, including component, node, user, queue count, type, primary and secondary master information.

Example 4-4 QUERY LTERM command output of consolidated LTERM resource information

```
Response for: QUERY LTERM NAME(*) SHOW(ALL)                        More:    >
Lterm    MbrName    CC    LQCnt LNode    Line Pterm LInCmp LOutCmp LUser
DFSTCF   IMSM        0        0 DFSLN004    4     1
DFSTCFI  IMSM        0        0 PMASTER1                      1
HOWARD   IMSM        0        0 USER1                  1      1
IVPPRT1  IMSM        0        0 DFSLN002    2     1
IVPSPL1  IMSM        0        0 DFSLN003    3     1
PMASTER  IMSM        0       22 PMASTER1               1      1
SMASTER  IMSM        0       41 DFSLN002    2     1
USER1    IMSM        0        0 USER1                  1      1
USER2    IMSM        0        0 USER2                  1      1
WTOR     IMSM        0        0 DFSLN001    1     1

(Page to the right within the TSO SPOC application to see a continuation of the
the output)

Response for: QUERY LTERM NAME(*) SHOW(ALL)                        More: <
Lterm    MbrName  Pterm LInCmp LOutCmp LUser          LVersion# LclStat
DFSTCF   IMSM        1                                        0 STATIC
DFSTCFI  IMSM                      1                          0 STATIC
HOWARD   IMSM               1      1                          0 STATIC
IVPPRT1  IMSM        1                                        0 STATIC
IVPSPL1  IMSM        1                                        0 STATIC
PMASTER  IMSM               1      1                          0 STATIC,MTO
SMASTER  IMSM        1                                        0 STATIC,SMTO
USER1    IMSM               1      1                          0 STATIC
USER2    IMSM               1      1                          0 STATIC
WTOR     IMSM        1                                        0 STATIC
```

QUERY NODE

Use the QUERY NODE command to display information about VTAM terminals represented by nodes, and non-VTAM devices (such as the system console, SPOOL, SYSOUT, and TCO devices) across the IMSplex. You can only specify this command through the OM API, which is also valid on an XRF alternate.

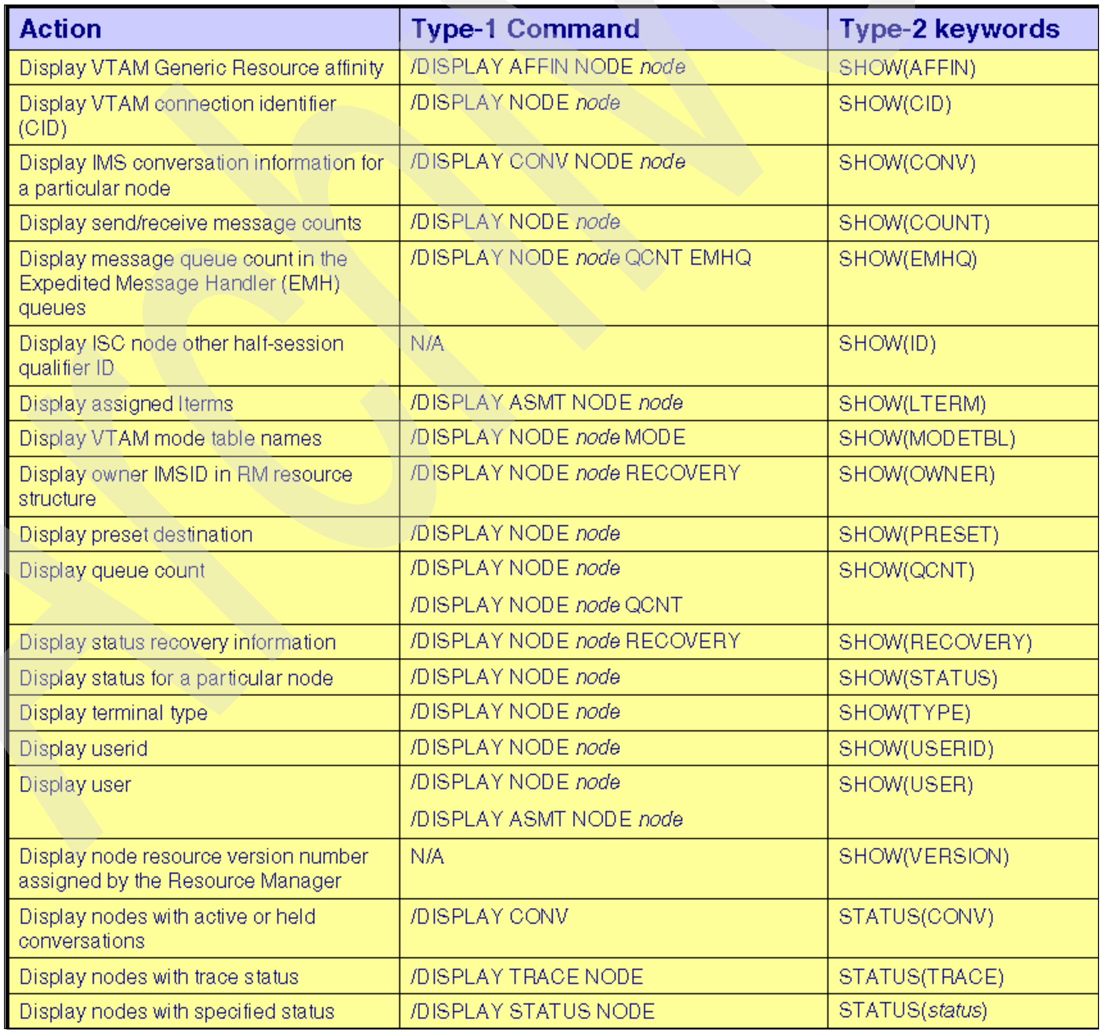
To display node resource information, issue the QUERY NODE command specifying one or more node names; or use a wildcard asterisk (*) character to represent all node names. Non-VTAM devices are referred to by the characters DFSLN followed by a line number, which represent the node name. You can filter the command output to show only the nodes that are associated with specified ISC user names. You can also filter the command output to show only nodes that have a particular status.

Prior to IMS 11, to retrieve information about nodes, you had to issue various type-1 /DISPLAY commands, such as:

- /DISPLAY NODE
- /DISPLAY AFFIN NODE
- /DISPLAY CONV
- /DISPLAY ASMT NODE
- /DISPLAY STATUS NODE

By issuing a single QUERY NODE command, you can display the same information that all these /DISPLAY commands can, and the ID of the other half-session qualifier for an ISC parallel session.

The table in Figure 4-3 summarizes the resource information that can be seen with one QUERY NODE command in place of several equivalent type-1 /DISPLAY commands.



Action	Type-1 Command	Type-2 keywords
Display VTAM Generic Resource affinity	/DISPLAY AFFIN NODE *node*	SHOW(AFFIN)
Display VTAM connection identifier (CID)	/DISPLAY NODE *node*	SHOW(CID)
Display IMS conversation information for a particular node	/DISPLAY CONV NODE *node*	SHOW(CONV)
Display send/receive message counts	/DISPLAY NODE *node*	SHOW(COUNT)
Display message queue count in the Expedited Message Handler (EMH) queues	/DISPLAY NODE *node* QCNT EMHQ	SHOW(EMHQ)
Display ISC node other half-session qualifier ID	N/A	SHOW(ID)
Display assigned lterms	/DISPLAY ASMT NODE *node*	SHOW(LTERM)
Display VTAM mode table names	/DISPLAY NODE *node* MODE	SHOW(MODETBL)
Display owner IMSID in RM resource structure	/DISPLAY NODE *node* RECOVERY	SHOW(OWNER)
Display preset destination	/DISPLAY NODE *node*	SHOW(PRESET)
Display queue count	/DISPLAY NODE *node* /DISPLAY NODE *node* QCNT	SHOW(QCNT)
Display status recovery information	/DISPLAY NODE *node* RECOVERY	SHOW(RECOVERY)
Display status for a particular node	/DISPLAY NODE *node*	SHOW(STATUS)
Display terminal type	/DISPLAY NODE *node*	SHOW(TYPE)
Display userid	/DISPLAY NODE *node*	SHOW(USERID)
Display user	/DISPLAY NODE *node* /DISPLAY ASMT NODE *node*	SHOW(USER)
Display node resource version number assigned by the Resource Manager	N/A	SHOW(VERSION)
Display nodes with active or held conversations	/DISPLAY CONV	STATUS(CONV)
Display nodes with trace status	/DISPLAY TRACE NODE	STATUS(TRACE)
Display nodes with specified status	/DISPLAY STATUS NODE	STATUS(*status*)

Figure 4-3 Type-1 /DISPLAY and type-2 QUERY NODE SHOW() equivalent commands

The syntax of the QUERY NODE command is shown in Figure 4-4.

```
QUERY NODE NAME(nodename1,nodename2,...)
USER(username1,username2,...)
SHOW(LOCAL | GLOBAL | attributes)
STATUS(status)
```

Figure 4-4 QUERY NODE command syntax

The parameters of this command are:

- NAME() is where you can specify any of the following items:
  - VTAM node names/IMS terminals
  - Non-VTAM devices (represented by “DFSLN” followed by a line number)
  - Wildcard asterisk (*) character (mutually exclusive with other parameter values)
- USER() is a filter where the ISC user name (or names) allocated to the node are specified (a wildcard asterisk (*) character is supported).
- SHOW() is used to specify the scope of what is to be shown in the command output. Both local and global NODE queue counts can be displayed, in addition to other NODE attribute values.
- STATUS() is used as a filter to display nodes that have a specific designated status.

Sample scenarios

As you can see, output from several types of /DISPLAY commands has been consolidated into output from a single QUERY NODE command. The examples in this section show how the output is consolidated.

Example 4-5 shows the DISPLAY for non-VTAM devices.

Example 4-5 /DISPLAY LINE command output (with line wrapping shown)

```
*996 DFS996I *IMS READY*
 R 996,/DIS LINE ALL
 DFS000I     LINE TYPE     ADDR    RECD ENQCT DEQCT   QCT  SENT
 DFS000I        1 CONSOLE  ****      38    24    24     0    88
 DFS000I        2 RDR/PTR  ****       0    56     0    56     0 NOOUT NOQUEUE
DEACT IDLE NOTOPEN
 DFS000I        3 RDR/PTR  ****       0     0     0     0     0 STOPPED IDLE
NOTOPEN
 DFS000I     *09198/220559*
*997 DFS996I *IMS READY*
```

Example 4-6 shows the DISPLAY for node affinity.

Example 4-6 /DISPLAY AFFIN NODE command output

```
*995 DFS996I *IMS READY*
 R 995,/DIS AFFIN NODE ALL
 DFS000I     NODE     APPLID
 DFS000I     *        N/A
 DFS000I     *09198/220347*
*996 DFS996I *IMS READY*
```

Example 4-7 shows the DISPLAY for conversation ID, transaction and status associated with nodes.

Example 4-7 /DISPLAY CONV command output

```
*998 DFS996I *IMS READY*
 R 998,/DIS CONV
 DFS000I     TERMINAL          USER     ID   STATUS
 DFS000I     NO CONVERSATIONS
 DFS000I     *09199/200039*
*999 DFS996I *IMS READY*
```

Example 4-8 shows the DISPLAY of LTERMs associated with nodes.

Example 4-8 /DISPLAY ASMT NODE command output

```
*999 DFS996I *IMS READY*
 R 999,/DIS ASMT NODE ALL
 DFS000I     NODE     USER     LTERM
 DFS000I     PMASTER1          I/O- PMASTER
 DFS000I     USER1             I/O- USER1   , HOWARD
 DFS000I     USER2 I/O- USER2
 DFS000I     *09199/200203*
*001 DFS996I *IMS READY*
```

Example 4-9 shows the DISPLAY of Node user ID, connection ID, message information, queue count, status, and type.

Example 4-9 /DISPLAY NODE command output

```
*994 DFS996I *IMS READY*
 R 994,/DIS NODE ALL
 DFS000I     NODE-USR TYPE   CID       RECD ENQCT DEQCT   QCT  SENT
 DFS000I     PMASTER1 3277   00000000     0    22     0    22  0 IDLE STATIC
 DFS000I     USER1    3277   00000000     0     0     0     0  0 IDLE STATIC
 DFS000I     USER2    3277   00000000     0     0     0     0   0 IDLE STATIC
 DFS000I     *09198/220249*
*995 DFS996I *IMS READY*
```

Example 4-10 shows the DISPLAY of node status.

Example 4-10 /DISPLAY STATUS command output

```
*006 DFS996I *IMS READY*
 R 006,/DIS STATUS NODE
 DFS000I     **NODE******
 DFS000I     PMASTER1  DISCONNECTED
 DFS000I     USER1     DISCONNECTED
 DFS000I     USER2     DISCONNECTED
 DFS000I     *09199/215323*
*007 DFS996I *IMS READY*
```

With IMS 11, the output shown in each of the previous command examples can be seen with a single type-2 QUERY NODE command. As you can see in Example 4-11 on page 106, all of the node information we obtained in the previous scenarios are listed and in a more clear format.

Example 4-11 QUERY NODE command output shows consolidated node resource information

```
Response for: QUERY NODE NAME(*) SHOW(ALL)                          More:    >
Node     Line Pterm MbrName    CC Gbl  Status    LQCnt LType        CID
DFSLN001    1     1 IMSM        0                    0 CONSOLE  00000000
DFSLN001            IMSM        0 Y
DFSLN002    2     1 IMSM        0                   66 RDR/PRT  00000000
DFSLN002            IMSM        0 Y
DFSLN003    3     1 IMSM        0                    0 RDR/PRT  08053726
DFSLN003            IMSM        0 Y
DFSLN004    4     1 IMSM        0                    0 TCO      00000000
DFSLN004            IMSM        0 Y
PMASTER1            IMSM        0                   24 3277     00000000
PMASTER1            IMSM        0 Y
USER1               IMSM        0                    0 3277     00000000
USER1               IMSM        0 Y
USER2               IMSM        0                    0 3277     00000000
USER2               IMSM        0 Y

(Page to the right within the TSO SPOC application to see a continuation of the
the output)

Response for: QUERY NODE NAME(*) SHOW(ALL)                          More: <
Node     MbrName  Gbl   RecdCnt  SentCnt LSRM LclStat
DFSLN001 IMSM                45       96      STATIC
DFSLN001 IMSM     Y
DFSLN002 IMSM                 0        0      STOLGN,DEACT,IDLE,STATIC
DFSLN002 IMSM     Y
DFSLN003 IMSM                 0        0      STOLGN,IDLE,STATIC
DFSLN003 IMSM     Y
DFSLN004 IMSM                 0        0      STATIC
DFSLN004 IMSM     Y
PMASTER1 IMSM                 0        0      IDLE,STATIC
PMASTER1 IMSM     Y
USER1    IMSM                 0        0      IDLE,STATIC
USER1    IMSM     Y
USER2    IMSM                 0        0      IDLE,STATIC
USER2    IMSM     Y
```

QUERY USER

Use the QUERY USER command to display information about users within an IMSplex, which can be either a dynamic (ETO) user or an ISC subpool user (static user or a dynamic user). To display user information, issue the QUERY USER command specifying one or more user names (or use a wildcard asterisk (*) character for all). You can filter the command output to show only the users that have a particular status.

Prior to IMS 11, to retrieve information about users, you had to issue various type-1 /DISPLAY commands, such as:

- /DISPLAY USER
- /DISPLAY ASMT USER
- /DISPLAY USER RECOVERY
- /DISPLAY STATUS USER

In the case of /DISPLAY USER, you could issue the command to display ETO users, ISC subpools, and security user IDs. With the enhancement to the QUERY command in IMS 11, you can issue this command to display a more granular user-level: issue the type-2 QUERY USER command to display ETO users and ISC subpools and issue the type-2 QUERY USERID command to display security user IDs (refer to “QUERY USERID” on page 108 for more detail).

The QUERY USER command consolidates output from several /DISPLAY commands previously listed, such as:

- Autologon information
- Conversation information
- Associated LTERMs
- Associated nodes
- Owning IMS system
- Preset destination name
- Recovery mode values
- Signed-on user ID
- Status of user

The syntax of the QUERY USER command is shown in Figure 4-5.

```
QUERY USER NAME(username1,username2,...)
SHOW(LOCAL | GLOBAL | attributes)
STATUS(status)
```

Figure 4-5 QUERY USER command syntax

The parameters of this command are:

- NAME() is where you can specify the (dynamic) ETO user (or users), and the ISC subpool users that should be displayed in the command output. A wildcard asterisk (*) character is supported here.
- SHOW() is used to specify the scope of what is to be shown in the command output. Both local and global users displayed, in addition to other user attribute values such as those listed previously that can be seen in various /DISPLAY commands.
- STATUS() is used as a filter to display users that have a specific designated status.

Sample scenarios

As previously mentioned, output from several /DISPLAY commands has been consolidated into output from a single QUERY USER command. We show how the output is consolidated with some examples.

Example 4-12 shows the display of user status, node, and user ID.

Example 4-12 /DISPLAY USER command output

```
*211 DFS996I *IMS READY*
 R 211,/DIS USER ALL
 DFS000I     USER     USERID   ENQCT DEQCT   QCT
 DFS000I     PAOLOR1  PAOLOR1      0     0     0 ALLOC(SC38E001)
 DFS000I     ANGIE    N/A          0     0     0 STOPPED
 DFS000I     *09201/180658*
*212 DFS996I *IMS READY*
```

Example 4-13 shows the display of user ID and node.

Example 4-13 /DISPLAY ASMT USER command output

```
*213 DFS996I *IMS READY*
 R 213,/DIS ASMT USER ALL
 DFS000I     USER     USERID   ID       NODE
 DFS000I     PAOLOR1  PAOLOR1           SC38E001 I/O- PAOLOR1
 DFS000I     ANGIE    N/A                        I/O- NONE
 DFS000I     *09201/182622*
*214 DFS996I *IMS READY*
```

Example 4-14 shows the display of user recovery options.

Example 4-14 /DISPLAY USER ALL RECOVERY command output

```
*214 DFS996I *IMS READY*
 R 214,/DIS USER ALL RECOVERY
 DFS000I     NODE-USR  OWNER    SRM    CONV STSN FPATH
 DFS000I     PAOLOR1   IMSM     LOCAL   Y    N     Y
 DFS000I     ANGIE
 DFS000I     *09201/182655*
*215 DFS996I *IMS READY*
```

Example 4-15 shows the display user status output.

Example 4-15 /DISPLAY STATUS USER command output

```
*216 DFS996I *IMS READY*
 R 216,/DIS STATUS USER
 DFS000I     **USER******
 DFS000I     ANGIE     STOPPED
 DFS000I     *09201/182810*
*233 DFS996I *IMS READY*
```

With IMS 11, the output shown in each of the previous command examples can be seen with a single type-2 QUERY USER command. As you can see in Example 4-16, all of the user information we obtained in the example scenarios are listed in a consolidated format.

Example 4-16 QUERY USER command shows consolidated user resource information

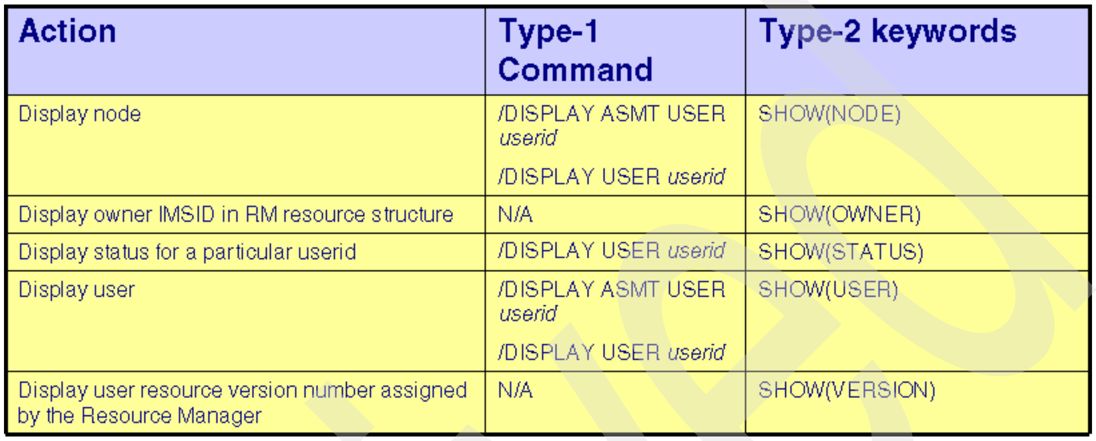
```
Response for: QUERY USER NAME(*) SHOW(ALL)
User     MbrName    CC    LQCnt LSRM LRcvy    LUserid  LNode    LclStat
ANGIE    IMSM        0        0                                 STOSGN
PAOLOR1  IMSM        0        0 LCL  CONV,FP  PAOLOR1  SC38E001 ALLOC
```

QUERY USERID

Use the QUERY USERID command to display RACF user IDs that are associated with the IMSplex that are used for user or terminal security. Note that the user ID we are referring to is different from the user resource in “QUERY USER” on page 106, which is a dynamic (ETO) or ISC subpool user. As with the other IMS 11 QUERY commands, QUERY USERID can be specified only through the OM API and is invalid on an XRF alternate IMS system.

Prior to IMS 11, to obtain information about RACF user IDs for terminal security, you could issue the /DISPLAY USER or /DISPLAY ASMT commands. However, a distinction is now drawn between the concept of a user versus user ID. Using these /DISPLAY commands, you could determine the node and user associated with the security user ID, and the user ID status. However, with the QUERY USERID command, you can see this same information as well as the owning IMS system and the version number associated with the user ID (when using a resource structure in each case).

Figure 4-6 summarizes the resource information that can now be seen with one QUERY USER command in place of several equivalent type-1 /DISPLAY commands.

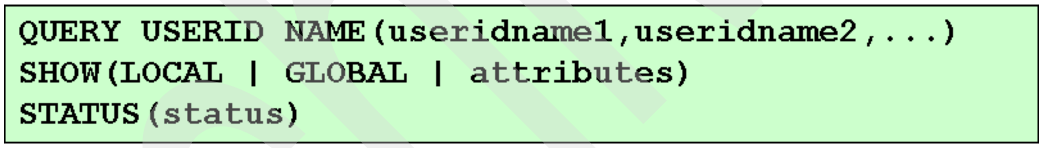


Action	Type-1 Command	Type-2 keywords
Display node	/DISPLAY ASMT USER *userid* /DISPLAY USER *userid*	SHOW(NODE)
Display owner IMSID in RM resource structure	N/A	SHOW(OWNER)
Display status for a particular userid	/DISPLAY USER *userid*	SHOW(STATUS)
Display user	/DISPLAY ASMT USER *userid* /DISPLAY USER *userid*	SHOW(USER)
Display user resource version number assigned by the Resource Manager	N/A	SHOW(VERSION)

Figure 4-6 Type-1 /DISPLAY and type-2 QUERY USERID SHOW() equivalent commands

Command syntax

The syntax of the QUERY USERID is shown in Figure 4-7.



```
QUERY USERID NAME(useridname1,useridname2,...)
SHOW(LOCAL | GLOBAL | attributes)
STATUS(status)
```

Figure 4-7 QUERY USERID command syntax

The parameters of this command are:

- NAME() can be one or more RACF user IDs that are to be displayed in the command output; a wildcard asterisk (*) character is supported.
- SHOW() can be used to display local or global user ID queue counts as well as specific attribute values.
- STATUS() is a filter used to display user IDs with a specific designated status.

Sample scenarios

As previously mentioned, output from different /DISPLAY commands has been consolidated into output from a single QUERY USERID command. We show examples of how the output is consolidated. Examples of the type-1 /DISPLAY command output are shown first for comparison to the type-2 QUERY USERID command output.

Example 4-17 shows the RACF security user ID associated user, node and status.

Example 4-17 /DISPLAY ASMT USER command output

```
*239 DFS996I *IMS READY*
 R 239,/DIS ASMT USER ALL
 DFS000I     USER     USERID   ID       NODE   IMSM
 DFS000I     IMS02    IMS02             SC38E002 I/O- IMS02
 DFS000I     IMS01    IMS01             SC38E001 I/O- IMS01
 DFS000I     ANGIE    N/A                        I/O- NONE
 DFS000I     *09202/143633*
*240 DFS996I *IMS READY*
```

Example 4-18 shows the RACF security user ID status and associated node.

Example 4-18 /DIS USER command output

```
*240 DFS996I *IMS READY*
 R 240,/DIS USER ALL
 DFS000I     USER     USERID   ENQCT DEQCT   QCT   IMSM
 DFS000I     IMS02    IMS02        0     0     0 ALLOC(SC38E002)
 DFS000I     IMS01    IMS01        0     0     0 ALLOC(SC38E001)
 DFS000I     ANGIE    N/A          0     0     0 STOPPED
 DFS000I     *09202/143650*
*241 DFS996I *IMS READY*
```

In IMS 11, a single QUERY USERID command shows the user and node name associated with the RACF security user ID, as shown in Example 4-19.

Example 4-19 QUERY USERID command shows consolidated user resource information

```
Response for: QUERY USERID NAME(*) SHOW(ALL)
UserID   MbrName    CC LNode    LUser
IMS01    IMSM        0 SC38E003 IMS01
IMS02    IMSM        0 SC38E004 IMS02
```

4.1.2 Operational considerations

The type-2 QUERY commands that are introduced in IMS 11 require the Operations Manager (OM) component of the Common Service Layer (CSL). The commands are entered through any OM interface, such as the TSO SPOC, IMS Control Center, Batch SPOC Utility or REXX SPOC. One IMS system processes this command, which is known as the *command master*. In an IMSplex environment, the IMS systems that are not selected by OM to be the command master can still participate in the command, showing their own local information. The command master IMS however, displays its own local information in addition to global information (in an IMSplex scenario).

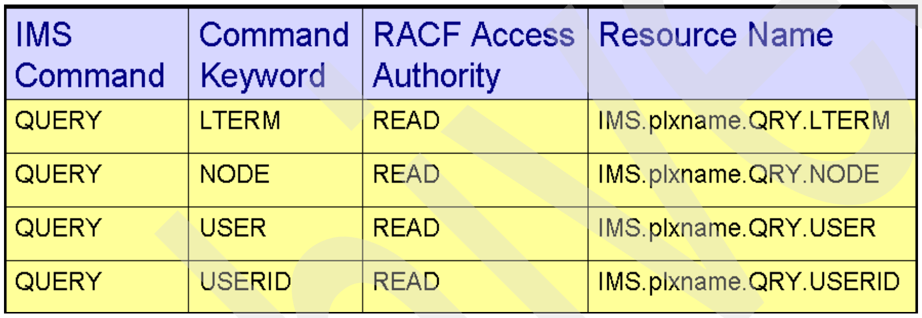
For more information about how to set up the CSL (including OM), refer to *IMS Version 11 System Definition,* GC19-2444. Note that you can issue these QUERY commands only on IMS systems that are at IMS 11. If you attempt to enter a command on an IMS system that is at an earlier release, the command is rejected. The addition of these commands does not affect existing type-1 commands or other types of the QUERY command that existed prior to IMS 11.

4.1.3 Performance considerations

Be careful when you issue any QUERY command with the NAME(*) parameter. If you include the NAME(*) parameter, which is the default, all resources of the specified type will be referenced. If you use a resource structure, the time taken to access all instances of a resource in it can be significant. In the same way, when including the MSGAGE parameter for applicable QUERY commands, each message on the shared queue is referenced for age determination. If performance is a concern, you can use an OM input user exit to prevent a QUERY command from being issue with NAME(*) parameter. This user exit can watch for the instance of a command with certain parameter values specified, such as NAME(*). When the exit detects this, it can modify the command in a way that you specify within the exit.

4.1.4 Security considerations

If you use RACF security, you have to define new RACF definitions so that the OM is able to issue the QUERY commands. Figure 4-8 summarizes these required definitions.



IMS Command	Command Keyword	RACF Access Authority	Resource Name
QUERY	LTERM	READ	IMS.plxname.QRY.LTERM
QUERY	NODE	READ	IMS.plxname.QRY.NODE
QUERY	USER	READ	IMS.plxname.QRY.USER
QUERY	USERID	READ	IMS.plxname.QRY.USERID

Figure 4-8 Required RACF definitions for type-2 QUERY commands in IMS 11

For more details about setting these RACF definitions, refer to *IMS 11 System Administration*, SC19-2443.

4.2 APPC enhancements

IMS 11 introduces a storage pool enhancement: IMS monitors both LUMP and LUMC storage pools to notify APPC/MVS when storage is approaching maximum limits. This enhancement is beneficial because shortages in storage pools can cause unpredictable results.

4.2.1 LUMP and LUMC storage pool monitoring

The monitoring provided for the LUMP and LUMC storage pools occur automatically when the storage limit for both pools is specified as greater than 8 MB. However, if the limit is set to 2 MB or if either pool’s limit is set below 8 MB, monitoring does not occur. Let us review the new DFS messages that are issued during monitoring.

DFS messages

When either of the LUMP or LUMC storage pool is approaching its limit, IMS issues one of the following messages and notifies APPC/MVS to accept no further IMS input (the value for `x` is either `C` or `P`):

- `DFS1277W LUMx POOL STORAGE SHORTAGE`

  This message is issued when the LUMP or LUMC storage pool has less than 5 MB of free storage remaining.

- `DFS1278E LUMx POOL LIMIT REACHED, ALL` **`APPC`** `INPUT WILL BE REJECTED`

  This message is issued when the LUMP or LUMC storage pool has less than 1 MB of free storage.

- `DFS1279I` **`APPC`** `INPUT PROCESSING RESUMED`

  This message is issued when the storage level for the involved pools decreases to an acceptable amount, and APPC/MVS is notified that IMS input can be resumed.

4.3 OTMA enhancements

IMS 11 introduces OTMA capabilities that improve consistency in the IMS shared queues environment, reduce overall transaction processing costs, and increase resiliency when certain conditions arise that if left untreated, could result in delays. In this section, we discuss how the capabilities provide these benefits.

4.3.1 Consistency enhancements for shared queues environment

This section discusses OMTMA capabilities to improve consistency in an IMS shared queues environment.

DFS555I message handling

In a shared queues environment, a common way to handle incoming messages is to designate certain IMS systems within the IMSplex as front-end or back-end systems. The front-end systems can receive transactions from terminals; back-end systems can receive transactions from a front-end IMS system through the shared queue, and handle processing.

A possibility is for a transaction to abend during processing, in which case a DFS555I message is issued, indicating that this has occurred. Prior to IMS 11, the OTMA client that sent the message transaction only received this message if a transaction abended on a front-end IMS system. If it abended on a back-end system, there was no knowledge of the OTMA client, so the client never received the message. This resulted in an OTMA client timeout handled on the client’s end, which was imperative, because there was no other mechanism available to break the connection and eliminate the hang.

IMS 11 introduces an enhancement that allows a transaction abend on a back-end IMS system to be handled in the same way as a front-end system. When an abend occurs on a back-end system, the DFS555I message flows to the OTMA client by way of the front-end system, eliminating the possibility of a timeout.

For details about the DFS555I message, refer to *IMS Version 11 Messages and Codes, Volume 1: DFS Messages,* GC18-9712.

Note: This enhancement was retrofitted to previous versions of IMS and is available through the service stream with the following maintenance (APAR/PTF):

- IMS Version 9: PK35745/UK31054
- IMS Version 10: PK38720/UK31057

Super member extension

If you use shared queues, your IMS Connect clients are now able to retrieve messages from an Alternate PCB (ALTPCB) that is generated by a back-end IMS system. Prior to IMS 11, an IMS Connect client was only able to retrieve messages that were already on the hold queue. Any messages that were subsequently put on the hold queue required another RESUME TPIPE request and were not automatically sent to the client. In this section, we discuss this issue and how IMS 11 alleviates it.

Overview of super member functionality

Prior to the introduction of super member, any ALTPCB created by a back-end IMS system was queued to the shared queue with affinity, specifying the back-end system's IMSID as part of the queue name. A remote IMS Connect client therefore had to know which back-end IMS system the RESUME TPIPE request should be associated with.

Super member introduced the capability of queuing ALTPCB messages to the shared queue, specifying the super member as the queue name instead of the usual IMSID-specific queue name. This approach enabled any front-end IMS to successfully process a RESUME TPIPE request sent from an IMS Connect client. Messages from the super member queue could therefore be retrieved, regardless of the back-end IMS that had queued them. Using a generic queue was the key that made this approach possible, but it applied only to the situation in which messages were already on the queue, either because they had not yet been sent or because of a previous send-failure. In other words, the only supported types of RESUME TPIPEs were NOAUTO, where all messages were sent per request, and SINGLE, where one message was sent per request.

Listener support added to super member

Messages that do not yet exist and would eventually be put on the queue require a different kind of RESUME TPIPE support: AUTO, where messages are sent as they arrive; and SINGLE-WAIT, where one message is sent upon arrival but a separate request is required for another send. If these particular types of RESUME TPIPEs are requested on versions prior to IMS 11, behind the scenes they would be converted to NOAUTO and SINGLE because the support for them had not yet been added.

With IMS 11, these types of RESUME TPIPE requests can be validly issued and a remote client can issue them to act as a listener for messages that are added to the shared super member queue.

Refer to *IMS Version 11 Communications and Connections,* SC19-2433 for more information about managing the retrieval of output messages using the RESUME TPIPE request.

Message processing with shared queues

If you are familiar with shared queues, you know that an IMS system registers interest in the shared queue that messages will eventually be queued to. Then, when messages are queued, the IMS that has previously registered interest is informed of the newly queued messages. The IMS is then capable of processing these messages and initiating a schedule in the dependent region.

The IMS 11 enhancement follows the same process in that the front-end IMS system registers interest in the super member TPIPE queue and any ALTPCBs that are generated at a back-end IMS system cause an inform to be issued to the front-end IMS system. This process is how the front-end IMS is made aware that a back-end system has generated output in an ALTPCB. The RESUME TPIPE request made by a remote client can then access these ALTPCB messages because of its connection to the front-end IMS system, which has access to the shared super member queue.

IMS Connect enhancement

IMS Connect has also been enhanced in IMS 11 to immediately time out a remote client when it receives a negative acknowledgement (NAK) after issuing a RESUME TPIPE request that did not have a reroute option specified with it. As we previously mentioned, the reroute function can be used for managing output generated by send-only transactions and for managing output that cannot be delivered to the original client because the connection timed out or failed.

If a RESUME TPIPE was requested by a client with a reroute specified and a timeout occurs, the request goes to a timeout queue instead of continuing to wait. Previously, the client would have been returned to a CONN state, making it unable to receive additional messages. This enhancement returns the client to RECV state after a timeout.

For more information about rerouting RESUME TPIPE requests, refer to the information about rerouting commit-then-send output in *IMS Version 11 Communications and Connections,* SC19-2433.

Rerouting commit-then-send output

You can configure IMS to reroute commit-then-send (commit mode 0) IOPCB output to an alternate OTMA TPIPE hold queue for retrieval.

Normally, if IMS cannot return commit mode 0 (CM0) output to the application client, the output is routed to the TPIPE hold queue associated with the client application that submitted the original message; however, if you request the reroute function, IMS reroutes the output to either a user-specified TPIPE hold queue or the default TPIPE hold queue HWS$DEF. Whether the reroute TPIPE is a user-specified TPIPE or the default TPIPE, the reroute TPIPE is always associated with the TMEMBER of the original TPIPE.

The reroute function can be used for managing output generated by send-only transactions and for managing output that cannot be delivered to the original client because the connection timed out or failed.

The reroute function is also useful when IMS TM resource adapter (formerly known as IMS Connector for Java) is used with shareable persistent sockets. The IMS TM Resource Adapter automatically generates the client ID when connecting to IMS Connect. Consequently, the client ID is unknown to the client applications, which requires the client ID to retrieve the CM0 output.

You can specify the rerouting function in either CM0 or CM1 input messages. However, in the case of CM1, IMS can only reroute the CM0 output, such as might be generated by a program-to-program switch.

Both user-written applications and IMS TM resource adapter applications on either persistent sockets or transaction sockets can request the reroute function.

Restrictions: The reroute function is not supported for:

- CM1 output messages
- Output resulting from a RESUME TPIPE call
- Output resulting from an insert to an ALTPCB

Advantages

Because a remote client now has the ability to listen for messages by issuing a RESUME TPIPE command indicating that messages should be sent as they are received, several benefits are realized. Using a shared queue with super member, any front-end IMS system can process a RESUME TPIPE request because of it having previously registered interested in the TPIPE shared queue. This process reduces the necessary effort required to rely on the remote client's end to make this possible. IMS Connect is the liaison between TCP/IP clients and IMS systems and now that message routing has been made more generic, its IP spraying capability has been enhanced.

4.3.2 OTMA transaction expiration and input message timeout

When IMS experiences a delay and an OTMA message arrives, the message might take too long to be processed, resulting in the client timing out. In this case, the client no longer requires a returned response after the message is processed. To resolve this issue, a capability has been introduced that allows you to specify an expiration time for each OTMA message. If this expiration time is reached while the message is waiting to be processed, the message is discarded with no attempt at processing. This approach prevents unnecessary delays that would otherwise occur while the client is waiting for a response.

Expiration value scope

You can control how long an OTMA message should wait before being processed instead of relying on the client to time out on its own in the case of an IMS or network hang. Any expiration value that you specify for an OTMA message overrides any other expiration time that was specified at the transaction level. In the latter case, the expiration value associated with a transaction applies to all messages queued to it. For more details about this other type of expiration value setting, refer to 4.5, "Transaction expiration" on page 138. The OTMA expiration value that you specify at the message level is available in IMS 11, and also in IMS 10 as a small program enhancement (SPE). The SPE is available through IMS APAR PK74017 and IMS Connect APAR PK74024. The transaction-level expiration value, however, is only available in IMS 11.

Implementation

You specify the expiration value in the OTMA message prefix in either a STCK (the *store clock* HLASM instruction that returns the current time in a specific format) time format or an elapsed time format (similar to when you specify a value at the transaction level). Use the STCK time format if you are using OTMA with an IMS Connect client. The other elapsed time format is not yet used with IMS Connect. We examine these two options in more detail, focusing on the STCK format.

STCK time format

IMS Connect is the mechanism that determines the STCK time that OTMA uses in its comparison to the current STCK time. For this to occur, you have to enable the transaction expiration function by setting the IRM_F1_TRNEXP value in the IRM_F1 field. This approach enables IMS Connect to use values that you specify either in its IRM_TIMER field of the IRM header or in the TIMEOUT value within the HWS configuration member to calculate the STCK time that is used in determining whether a message should be expired or not.

Note: As with OTMA, IMS Connect also uses messages to communicate with IMS, and are known as IMS request messages (IRMs). Each IRM begins with a header segment containing a 28-byte fixed-format section that is common to all messages from all IMS Connect client applications that communicate with IMS TM. One of the values within this header is IRM_TIMER, which tells IMS Connect how long to wait for IMS to return data before sending a NAK to the client.

IMS Connect uses either the IRM_TIMER or TIMEOUT values to calculate the STCK time that OTMA will use in determining whether a message is expired. After it has determined the STCK time, IMS Connect stores it in the User Data section of the OTMA prefix. When OTMA compares this stored STCK time value with the current STCK time, and the result is that the stored value is less than the current value, OTMA expires the message and it is discarded.

Within the State Data portion of the prefix, IMS Connect sets the flag TMAMTXP1 (x'01') in the TMAMHIST byte signifying that STCK format is being used (versus elapsed time format). Finally, it also sets a value in the TMAMOSXP field indicating the offset of the STCK time value contained in the User Data portion of the prefix. All remote clients that send OTMA messages to IMS are required to have these values set in both the User and State Data portions of the prefix.

The benefit of using IMS Connect to calculate the transaction expiration value using either the IRM_TIMER (in the IRM header) or TIMEOUT (in the HWS configuration member) values is that coordinating these values is now more straightforward. It now takes less effort to make these values as similar as possible. If the values were different from each other, a message could be expired on the IMS Connect end, but still valid on the IMS end. This could result in a message schedule followed by an unnecessary return of output to the client. However, when these values are coordinated, this mismatch is less likely and an expiration timeout is likely to occur in both IMS Connect and IMS, rather than in one and not the other.

Elapsed time format

The second format for the expiration value you can specify in the OTMA prefix is the elapsed time format. This format is not used with IMS Connect and is intended for future use, so we only briefly discuss the setup here. If you are using this format, you have to specify the TMAMTXP2 (x'02') flag within the TMAMHIST byte in the State Data portion of the OTMA prefix. Additionally, you have to set the TMAMOSPX field to the elapsed time value that OTMA must use in determining whether it should expire the message.

OTMA transaction expiration process

The three instances in which OTMA checks an input message and determines whether it has expired and should be discarded are:

- During the input-receiving phase: OTMA creates a time stamp after receiving an input transaction message from XCF, which it then compares to the expiration time. If OTMA detects that the message has expired, it issues a NAK to the client with sense code x'0034' with OTMA reason code x'0001'.
- During the enqueue phase: OTMA checks the expiration time before enqueuing an input transaction message. If it finds that the message has expired, it issues a NAK to the client with sense code x'0034' with OTMA reason code x'0002'.
- When an IMS application program issues a Get Unique (GU) call for an input transaction message, OTMA checks the expiration time. If it finds that the message has expired, an abend 0243 is issued with a DFS555I message. In a shared queues environment, the abend is issued at the back-end IMS system with a DFS2241I message.

If OTMA detects that the message has expired at any of the checks, it expires the message and then discard it. A x'67D0' log record is also written.

Note: Messages for the following transaction types are not checked for expiration during the GU phase (only during receiving and enqueuing phases):

- MSC remote
- IFP
- IMS conversational
- Switched-to for program-to-program switches

Error scenario

When OTMA receives a message through XCF, it checks the State Data portion of the prefix to determine whether you have set either of the transaction expiration flags (TMAMTXP1 and TMAMTXP2) validly. If you have set both flags, OTMA issues a NAK to the client with sense code x'001A' along with a new OTMA reason code x'0065' indicating this invalid situation. Only one of these two flags can be set to request the OTMA transaction expiration function.

Considerations

Message-level expiration specification is optionally implemented by an OTMA client and is supported by IMS Connect. If both message level and transaction specification levels are simultaneously used to handle expiration for input transaction messages, the message-level specification takes precedence. Otherwise, transaction level specification is used. Refer to 4.5, "Transaction expiration" on page 138 for details about this method of expiration specification.

Benefits

Input transaction messages handled by OTMA are now managed with a higher level of efficiency because they are not being scheduled unnecessarily if they are no longer required by the client. With the transaction expiration enhancement, you now have the flexibility of specifying an expiration value at either the transaction level (applies to all associated messages) or with higher granularity with the message level specification-override capability. If this expiration time is reached while the message is waiting to be processed, the message is discarded with no attempt at processing. This approach prevents unnecessary delays that would otherwise occur while the client is waiting for a response.

When using transaction-level specification, you are not required to make any changes to your OTMA members or clients to take advantage of the capability. When using message-level specification, IMS Connect handles setting the expiration value for you by using its IRM header timer capability. It also simplifies the coordination of different timeout values used by IMS Connect and OTMA.

4.3.3 Enhanced timeout for hung TPIPEs

IMS 11 extends the IMS 10 timeout support by including applicability to OTMA Commit Mode 0 (CM0) messages, in which changes made by an IMS application program are committed before a response is sent to the OTMA client. In this case, the messages are not dequeued from the TPIPE until an ACK from the client is received. In IMS 10, timeout support only existed for commit mode 1 (CM1) messages that were issued with `synclevel=confirm` or `synclevel=syncpt` values, in which an ACK is required from the client before application program changes can be committed.

An ACK timeout value is useful in preventing IMS from unnecessarily holding resources during the syncpoint process when a ACK/NAK is not received because of a network failure or delay, or a client programming error. We now discuss the mechanics behind this capability.

Implementation

A TPIPE is considered hung when IMS is waiting for an ACK/NAK for an extended period of time and as a result, messages can no longer be delivered to the OTMA client. With the ACK timeout capability in IMS 11, CM0 messages are monitored to automatically detect this situation and take corrective action. When OTMA detects an ACK timeout, it moves the waiting message to a different queue so that other messages can be sent to the OTMA client with this TPIPE.

Process to free a hung TPIPE

In the case of a non-hold queue client, when OTMA detects that an ACK has timed out, it attempts to move the CM0 output message to the queue that was specified during the OTMA client-bid time. If you did not specify a TPIPE name to be used in the client-bid process, the message is moved to the DFS$$TOQ TPIPE, which is the default timeout queue. An example of a non-hold queue client is a non-IMS Connect client.

Note: A client-bid occurs when an OTMA client joins an XCF group and requests a connection to the IMS server.

Typically, hold queue clients are treated the same as non-hold queue clients, except when an IOPCB output message is being handled. Similar to the non-hold queue client case, a CM0 message (only on a hold queue) that has an ACK timeout is moved to the timeout TPIPE specified during the client-bid process; if you did not specify one, it is moved to the DFS$$TOQ TPIPE. The DFS3494E message is also issued. In the case of an IOPCB output message, OTMA first determines whether the message was originally sent to IMS with a reroute TPIPE option specified. If it was, OTMA moves the message to that TPIPE. If the original input message was not specified with a reroute option, OTMA follows the course of action that it takes for a CM0 message as previously described.

In all of the previously described ACK timeout scenarios, a DFS3494E message is issued to both the system console and to MTO, indicating that an ACK timeout has occurred. For details about this message, refer to *IMS Version 11 Messages and Codes, Volume 1: DFS Messages,* GC18-9712-02.

Designating the timeout TPIPE to be used during OTMA client-bid

To use a timeout TPIPE other than the default DFS$$TOQ TPIPE, you can specify it within the State Data portion of the OTMA user prefix. Set the TMAMHFG2 client-bid flag to TMAMSTO (which equates to x'04') to indicate that OTMA provide the timeout TPIPE name that is set during the client-bid process. Next, set the 8-byte TMAMTOQN field (under the TMAMTO timeout value) to TPIPE name you want to use. This TPIPE saves CM0 messages when an ACK timeout occurs.

In the case of IMS Connect, a CM0ATOQ parameter has been added to the HWS statement and the DATASTORE statement. This parameter represents a queue name and is 1 - 8 characters long. It is used to save CM0 messages in the event of an ACK timeout. If you specify this parameter on the HWS statement, then during the client-bid process, the timeout TPIPE name is passed on to all data stores associated with the IMS Connect instance. If you specify it on the DATASTORE statement, the HWS statement is overridden and the TPIPE name is sent only to that specific data store during the client-bid process.

When a CM0 message is put on a hold queue, an IMS Connect client can later retrieve it by issuing a RESUME TPIPE request specifying the timeout TPIPE name. At this point, the client should send an ACK to OTMA to prevent the timeout condition from being triggered again.

4.3.4 Resource monitoring

IMS 11 includes a monitoring function for OTMA message flood conditions, which can arise from a variety of reasons. A remote client can send in a large number of messages, or IMS can experience processing delays because of RACF, Queue Manager, or internal problems. In any of these cases, IMS can have so many messages waiting that it lacks the necessary resources required to process incoming requests. OTMA resource monitoring allows clients to be notified before a flood condition arises and before failures occur. This approach gives OTMA members an opportunity to react in a way that would prevent delays in message processing (or expiring if you are using the expiration function), such as rejecting further client messages or rerouting the messages to a less busy IMS system.

This function is retrofitted to IMS 10 by APARs PK70458 and PK70960.

Early warning and detection of a flood condition

Prior to IMS 11, OTMA had the capacity to detect a large number of unprocessed messages waiting to be processed in an IMS system. This would often prevent additional messages from being sent to the IMS system, exacerbating the situation. OTMA tracked these messages by creating an internal control block known as a transaction instance block (TIB) for each message received from an OTMA client. When a maximum number of TIBs was reached, any new messages from an OTMA member were rejected. This would cause an increase in LSQA storage, which would often result in a S40D IMS abend.

In IMS 11, OTMA continues to detect resource shortage conditions that exist within an IMS system when it has a high volume of work. It has been enhanced with an early flood-detection capability that sends warnings to an OTMA client before the input message threshold has been surpassed. This capability exists both at the individual OTMA client level and also at a global level, which applies to all OTMA clients.

Implementation

Let us explore what happens when OTMA is monitoring for imminent message-flood conditions. Three phases are involved in the early detection process: the client-bid phase, the processing phase, and the client-disconnect phase.

Client-bid phase

In this phase, an OTMA client connects to the IMS server and OTMA starts its internal monitor that sends a message to IMS at regular intervals indicating the state of the resources that this client requires. If an OTMA client is only reconnecting to the IMS server, the current resource information is returned during the client-bid process.

Processing phase

After the client-bid process has completed, OTMA sends a *heartbeat message* to the client every 60 seconds containing resource status and availability information. If a severe issue occurs, a message known as an *action message* is immediately sent to IMS. Examples of a severe issue include an IMS abend or the total messages having reached 100% of the message threshold. These two message forms that OTMA sends to the clients are known as *OTMA protocol messages* and have a specific format, which we discuss in "OTMA protocol message: heartbeat and action" on page 120. After these imminent flood conditions subside, OTMA sends another message to the client indicating that the levels have decreased to an acceptable level.

Client-disconnect phase

When an OTMA client disconnects from the IMS server, OTMA does not stop monitoring the resources. Instead, it keeps track of the availability of the resources so that when the client reconnects to the IMS server, the information can be readily available and sent to the client upon reconnect.

Use cases

When might the OTMA early message flood detection capability be useful? The protocol messages that OTMA sends to the client, when a certain message threshold is triggered, contain information that can be used by an IMS Connect user exit or by various vendor products. These exits or other products can use the information contained in the header to take certain desired actions, such as halt message- sending to the delayed IMS, either permanently or temporarily until the percentage threshold of messages has decreased, or simply continue to monitor the status of the resources without taking any other action.

Let us say we have an IMS Connect as an OTMA client that is sending transaction input messages to IMS. The OTMA resource monitor is keeping track of the number of messages currently queued in IMS that are awaiting processing, and it is sending a heartbeat message back to the IMS Connect client every 60 seconds. Then, IMS begins to experience delays and IMS Connect keeps sending transaction input messages to the point that OTMA detects that 80% of the message threshold has been reached. OTMA then sends an action message to IMS Connect, informing it of the imminent message flood situation. An IMS Connect user exit or vendor product can monitor for this condition and handle it accordingly, for example, by preventing new messages from being sent to IMS.

In another example, let us say we are using two OTMA clients such as IMS Connect and WebSphere MQ (also known as MQ or MQSeries®). Both clients are sending transaction input messages to IMS and the threshold is reached, so the OTMA resource monitor sends an action message to each client. At this point, either or both of the clients can react to the messages as appropriate. The heartbeat messages continue to be sent to the clients so that they can each be kept abreast of the flood condition and whether or not it has been relieved. In addition to resource-availability information, notifications for severe conditions such as an IMS abend are also sent to these OTMA clients.

OTMA protocol message: heartbeat and action

The OTMA protocol message contains information pertaining to resource- availability status, also bit maps that apply to messages that are in a state of alert as a result of having reached a high-message volume. Let us now examine the format of this message further.

The protocol message is sent to OTMA clients in the form of a server-state protocol command and is located at byte four of the OTMA control data prefix. When x’3C’ is contained in the byte, it signifies to the OTMA client that the message is providing resource availability information. OTMA issues the server state protocol command in the following circumstances:

- When an OTMA client establishes a TPIPE connection
- When a significant change occurs in the ability of IMS to process OTMA messages, such as an IMS abend
- As a heartbeat message at 60 second intervals

Within the server state protocol message, the overall state of an IMS system is categorized as either normal (available), degraded, or unavailable:

- A normal state indicates that IMS is available and is processing OTMA messages normally.
- A degraded state indicates that IMS is processing OTMA messages more slowly than expected. OTMA issues a degraded state protocol command when one or more conditions indicate that IMS is not processing OTMA messages as quickly as it should.
- An unavailable state signifies that IMS can no longer accept OTMA transactions for processing. OTMA issues the unavailable state protocol command to alert the OTMA client that one or more severe conditions prevent IMS from processing OTMA messages.

Each state is represented within the protocol message, within a specific byte, which we show in Table 4-1.

In addition to notifying the client of the overall state of IMS processing, if the IMS processing is in either a degraded or unavailable state, the server state protocol command can provide information about the specific resources that are causing the overall state to be considered degraded or unavailable. This information is contained in two bit maps residing within the protocol message, one for degraded resources and another for unavailable resources.

The OTMA message header contains a control section that details the information in the prefix. When a value of x'3C' is present in this control section, it signifies that a protocol message is present within the state data section of the OTMA header. Table 4-1 summarizes the information contained within a protocol message.

Table 4-1 OTMA protocol message contents

Byte	Length	Content
0	2	Length of the state data for protocol message (80 bytes)
2	2	Overall status code: ► 3: Available for work for this member ► 2: Degraded; one or more warning conditions ► 1: Unavailable for work; experience one of more severe conditions
4	4	Bit map for OTMA resources that are in the severe state
8	4	Bit map for OTMA resources that are in the warning state
12	1	Status flag: x'80' means that this is a regular heartbeat message, which is issued every 60 seconds; otherwise, this is an action message
13	3	Reserved
16	16	OTMA server name
32	16	OTMA client XCF member name
48	20	Reserved
68	12	UTC time

As introduced in "Processing phase" on page 119, when the OTMA protocol message is sent to the client, it can be either a regular heartbeat message or an action message. A heartbeat message is sent at 60-second intervals to keep the OTMA client informed of resource availability; an action message is sent to the client when IMS abends or the message quantity reaches a specified threshold. Byte 12 of the protocol message contains a flag indicating which of these two message types are being sent. If this flag is on (set to x'80'), the message is a regular heartbeat message; otherwise, it is an action message catalyzed by a resource-availability state-change or IMS abend.

As we know, the OTMA protocol message contains overall status information for the IMS system and bit maps with information about degraded or unavailable resources. The resource information contained within these bit maps can be specific to one OTMA member, or it can pertain to a shared global resource that would apply to all OTMA members. As such, the bit maps have two subsections, one for local resources and one for global. Figure 4-9 illustrates the layout for the protocol message contents.

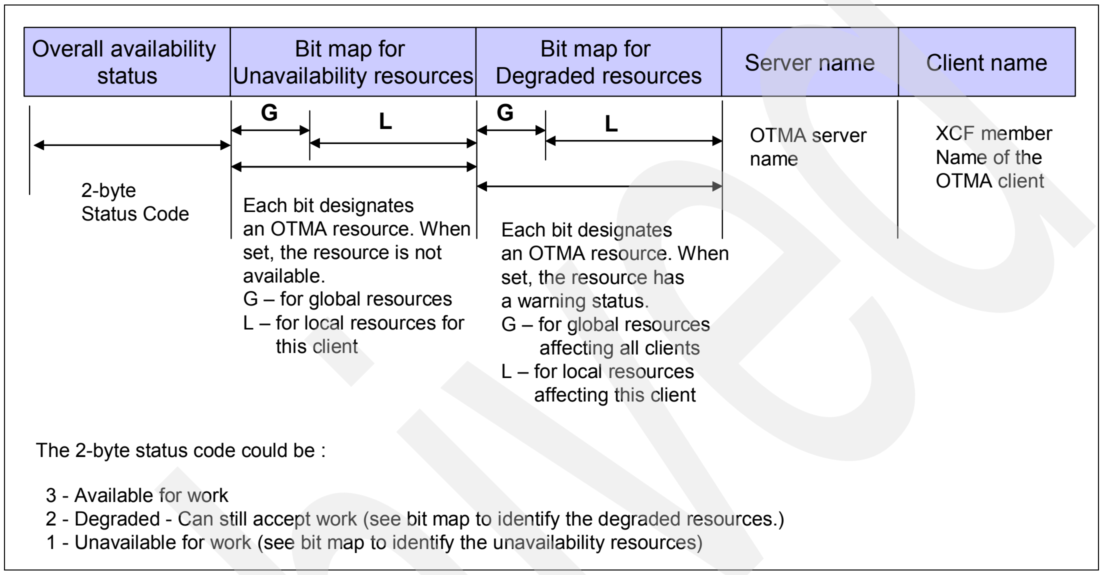

Overall availability status
Bit map for Unavailability resources
Bit map for Degraded resources
Server name
Client name
G
L
G
L
2-byte Status Code
Each bit designates an OTMA resource. When set, the resource is not available.
G – for global resources
L – for local resources for this client
Each bit designates an OTMA resource. When set, the resource has a warning status.
G – for global resources affecting all clients
L – for local resources affecting this client
OTMA server name
XCF member Name of the OTMA client
The 2-byte status code could be :
3 - Available for work
2 - Degraded - Can still accept work (see bit map to identify the degraded resources.)
1 - Unavailable for work (see bit map to identify the unavailability resources)

Figure 4-9 OTMA protocol message information layout

The bit maps can be used to assist you in determining the cause of the overall degraded condition. The resources that are flagged within these bitmaps can direct you to more closely examine a particular resource to troubleshoot its issue, thereby alleviating the overall degraded status. The OTMA server name and XCF member name of the OTMA client are also provided within the message.

Enhanced global command

OTMA is able to set a global message threshold for all OTMA members, which keeps track of the total number of messages that every OTMA member's clients are sending in to IMS. The default threshold value for a single OTMA member instance is 5000, and for all members combined, it is 8000. IMS 10 added the capability of specifying an override value for the default threshold of the individual OTMA member's message by adding an INPUT parameter to the /START TMEMBER command. IMS 11 expands this capability to apply to all OTMA members by adding the ALL parameter to this command.

The format of the command is:

```
/START TMEMBER ALL INPUT <value>
```

The command's INPUT value, whose maximum value can be 9999, overrides the global message threshold of 8000. If the total number of messages among all OTMA members reaches your specified value, a DFS4388W message is issued to the IMS MTO and system console. In addition, an OTMA protocol action message is sent to all OTMA clients. OTMA continues to monitor the total message volume and when it decreases to an acceptable level, a DFS0798I message is issued to the IMS MTO and system console. An updated protocol message is sent to all OTMA clients too, indicating that the high message volume has been relieved.

As part of this enhancement, the output for the /DISPLAY OTMA command has been enhanced to include the global message flood-warning level among the other OTMA server information, which is displayed under a column labeled INPT. Example 4-20 shows the command output.

Example 4-20 /DISPLAY OTMA command output showing the INPT column

```
GROUP/MEMBER XCF-STATUS USER-STATUS    SECURITY TIB INPT SMEM
              DRUEXIT  T/O
HARRY
-IMS1        ACTIVE     SERVER          FULL
-IMS1         N/A
-HWS001      ACTIVE     ACCEPT TRAFFIC   FULL    0   0   SM01
-HWS001       HWSYDRU0   5
-SM01                   SUPER MEMBER                     SM01
-SM01         N/A
-MQS001      ACTIVE     ACCEPT TRAFFIC   FULL    0   0
-MQS001       MQSYDRU0  10
```

OTMA messages

Several instances exist in which IMS and OTMA send notifications to indicate that a message flood condition might be imminent. These notifications include various DFS messages and OTMA protocol messages that are catalyzed in different ways, described in this section.

Enhanced OTMA messages

Prior to IMS 11, OTMA issued DFS messages, related to imminent flood conditions, to the system console only. The following messages are now issued to the system console, and also to the IMS MTO and the audit log, so that an Automated Operator Interface (AOI) exit can monitor for these conditions:

- `DFS1988W OTMA INPUT MESSAGES FROM MEMBER yyyy HAVE REACHED xx% OF THE MAX CONCURRENT INPUT MESSAGE LIMIT zzzz`
- `DFS1989E OTMA INPUT MESSAGES FROM MEMBER yyyy HAVE REACHED THE MAXIMUM CONCURRENT INPUT MESSAGE LIMIT zzzz`
- `DFS0767I OTMA MESSAGE FLOOD CONDITION HAS BEEN RELIEVED FOR MEMBER yyyy`
- `DFS2386I OTMA IS CONNECTED TO MEMBER xxxxxxxx` (available in IMS 11)

DFS messages issued with OTMA protocol messages

When an individual OTMA member approaches the maximum message volume threshold, the following DFS messages are issued to the system console (as WTO messages), IMS MTO and audit log:

- `DFS4380W OTMA XCF MESSAGES FROM membername HAVE REACHED 80% OF THE MAXIMUM MESSAGE LIMIT xxxxxxxx`

  OTMA monitors its message volume as it receives messages from a client through XCF. This message is issued when an individual OTMA member detects that it has received at least 80% of its message threshold, although processing continues. An OTMA protocol message is sent back to the client to allow the rerouting of further transaction messages to a different IMS system. These messages can still be processed, but only after the delay in the IMS system is alleviated. Note that if the transaction expiration function is active, these messages that are waiting to be processed can expire because of the slowdown.

- `DFS4381I OTMA XCF MESSAGE FLOOD CONDITION HAS BEEN RELIEVED FOR MEMBER membername`

  When the message volume for an individual OTMA member has reached at least 80% of the maximum threshold and then subsequently decreases to the 50% level, this message is issued to indicate that the imminent flood condition is no longer present. An updated OTMA protocol message is sent to the client with an updated overall status for the resources the member requires.

- `DFS4388W TOTAL OTMA SEND-THEN-COMMIT(CM1) MESSAGE BLOCKS HAVE REACHED OR EXCEEDED WARNING LIMIT OF nnnn`

  This global message is sent to all clients of OTMA members when the message volume for CM1 (send-then-commit) messages has reached 100% of the global threshold of *nnnn* for the total messages among all OTMA members. The default *nnnn* value is 8000, but can be set to a different value by using the following command (as indicated in "Enhanced global command" on page 122).

  ```
  /START TMEMBER ALL INPUT <value>
  ```

  When the message is issued, processing continues but might experience a delay in being processed. Note that if the transaction expiration function is active, these messages that are waiting to be processed might expire because of the slowdown. An OTMA protocol warning message, indicating this situation, is also sent to all clients, enabling the clients to reroute further transaction messages to a different IMS system.

- `DFS0798I TOTAL OTMA UNCOMPLETED SEND-THEN-COMMIT(CM1) MESSAGE BLOCKS ARE DECREASING BELOW` nnnn

  This global message is sent to all clients of OTMA members when the total CM1 (send-then-commit) message volume for all OTMA members has reached 100% of the threshold of *nnnn* and subsequently decreases to at least 80% of the threshold value. An updated OTMA protocol message is sent to all members' clients with an updated overall status for the resources that the members require.

Note: All of these DFS messages are unsolicited, so if you use the Operations Manager (OM) Audit Trail to track your IMSplex activity, these messages are included in it. Refer to *IMS Version 11 Operations and Automation,* SC19-2441 for more information about this capability.

Action points for issuing DFS and OTMA protocol messages

The conditions under which the new DFS messages and OTMA protocol messages are issued are summarized in Figure 4-10 on page 125. The OTMA protocol messages in the figure are signified by `3C`, because byte 4 of the OTMA control data prefix contains x'3C' whenever OTMA is sending resource availability information.

These protocol messages are issued to the client in the format previously discussed in "OTMA protocol message: heartbeat and action" on page 120.

Member	Default Threshold	80%		(at 5% incr) 85% - 95%		100%		Relief		Shutdown	
		WTO MTO Auditing	3C	WTO MTO Auditing	3C	WTO MTO Auditing	3C	WTO MTO Auditing	3C	WTO MTO Auditing	3C
Individual Member Level	5000	X	X	X		X	X	at 50% X	X		
Global Level - All Members	8000	X	X	X		X	X	at 80% X	X	X	X

Figure 4-10 OTMA resource monitor action points

The figures shows the local (individual) and global scopes of message thresholds and subsequent handling. We can see that the default message threshold volume for individual OTMA members is 5000, and for all members it is 8000.

In the case of an individual OTMA member, when 80% of the specified message threshold is reached, both DFS and protocol messages are issued as a warning. This action also occurs at the 100% level as an indicator of a severe condition. For the levels of 85 - 95%, only DFS messages are issued each time the percentage increments by 5%. After the message volume decreases to at least 50%, another DFS message is issued and a protocol message is sent to the client, indicating that the situation has been relieved.

When message volumes are being monitored at the global level for all OTMA members, DFS and protocol messages are only sent when 100% of the threshold value has been reached. After the message volume decreases to at least 80%, another DFS messages are sent and a protocol message is sent to the client indicating that the situation has been relieved.

When you issue a /CHECKPOINT FREEZE command (or another command that would result in IMS shutting down), OTMA sends a global protocol message to all OTMA member clients to indicate that the IMS is no longer available because it is in shutdown mode. The bit map for unavailable resources within the protocol message contains an indicator for this particular reason.

IMS Connect support

IMS Connect has been enhanced to interpret the protocol messages that it receives from OTMA and use the information to update its data store table entry, which represents its connection to an IMS system. This information represents the overall resource availability for a particular IMS system. So, in turn, various vendor products can be enhanced to take certain actions based on the information contained in the data store entry. For example, IMS Connect can choose to route subsequent messages to another IMS system that has higher availability after learning that one IMS system has either degraded or has unavailable status.

Exit interface block data store expansion

For connections to IMS TM, IMS Connect keeps track of the status of IMS datastores in entries in the exit interface block data store (XIBDS). In IMS 11, this block has been expanded to include:

- A status flag indicating the overall state of the OTMA server, which can be:
  - x’03’ available and has a normal state
  - x’02’ has resources that are in warning state with degraded availability

  - x’01’ has resources that are in a severe state, or are unavailable
  - x’00’ has no status currently available
- A 4-byte bitmap for each set of resources with a degraded (warning) or unavailable (severe) state
- A 12-byte time stamp indicating when the last resource that was updated was received from OTMA

  The time stamp is updated every 60 seconds with OTMA heartbeat protocol message or when the OTMA server’s resources’ status changes

IMS Connect Event Recorder exit routine (HWSTECL0) enhancement

IMS Connect can be customized to pass event data to HWSTECL0, which stores all trace and event notifications through a recording routine and can be used by any event recording function. Examples of recordable data include:

- TCP/IP read and write
- RACF calls
- OTMA send and receive
- User exit calls
- Session errors
- Two-phase commit events

Each of these items is referred to as an event, and has an event number associated with it, which represents the specific event type. The purpose of the HWSTECL0 exit is to enable vendor applications to keep track of IMS Connect activities and take actions based on events. IMS 11 adds event number 45, which indicates that an event has been recorded in the XIBDS entry. The entry identifies OTMA resources that are in a severe, warning, or normal condition. This event occurs after IMS Connect receives a protocol message containing a resource status update. For more information about IMS Connect events, refer to *IMS Version 11 Exit Routines,* SC19-2437.

Display commands enhancement

You are now able to see the current state of a targeted IMS system when you issue the following commands, which display IMS Connect information:

- VIEWHWS
- VIEWDS
- QUERY MEMBER
- QUERY DATASTORE

These IMS Connect commands now have an output field (`STATE=`) included in the command response on the TARGET MEMBER output line; the values and their meanings are summarized in Table 4-2.

Table 4-2 STATE values for IMS Connect display commands

STATE= value in display output	Meaning
AVAIL	The OTMA server is available
WARN	The OTMA server has one or more degraded resources and is in a warning state
SEVERE	The OTMA server has one or more severely affected resources and is in a severe state
N/A	No data store status information is available

Example 4-21 shows the `STATE=` field in the command output after the following command has been issued:

```
F HWS1,QRY MEMBER TYPE(IMSCON)
```

Example 4-21 The IMS Connect QUERY MEMBER command showing the STATE output field

```
HWSC0001I DATASTORE=IMS1 STATUS=ACTIVE
HWSC0001I GROUP=XCFGRP1 MEMBER=HWS1
HWSC0001I TARGET MEMBER=IMS1 STATE=AVAIL
HWSC0001I DEFAULT REROUTE NAME=CFG$DEF
HWSC0001I RACF APPL NAME=APPLID1
HWSC0001I OTMA ACEE AGING VALUE=2147483647
HWSC0001I OTMA ACK TIMEOUT VALUE=120
HWSC0001I OTMA MAX INPUT MESSAGE=5000
HWSC0001I SUPER MEMBER NAME=SM01
```

Migration

Refer to Chapter 7, “Installation and migration considerations” on page 227 for more information about migration considerations for the OTMA Resource Monitor.

4.3.5 Message processing buffer pool

IMS 11 has a dynamic storage buffer pool that is used by both APPC and OTMA for message processing. Refer to 4.9, “Dynamic storage private buffer pool enhancement” on page 146 for more information.

4.4 OTMA type-2 commands

IMS 11 introduces four type-2 commands that help you to manage your OTMA environment. These commands provide information about message volumes and associated transaction instances, which allow you to take action to prevent potential problems from arising because of high-message volumes. They also provide a more flexible way to manage callout requests from IMS. In addition, by allowing you to dynamically create and update OTMA descriptors, the addition of these commands increases IMS availability, by eliminating the requirement for an outage when adding, changing, or deleting OTMA descriptors.

Because these are type-2 commands, you have to issue them from an Operations Manager (OM) interface, such as the IMS TSO SPOC application or GUI-based IMS Control Center. Refer to *IMS Version 11 Operations and Automation,* SC19-2441 for more information about each of these interfaces. OM is a component of the Common Service Layer (CSL), which has to be in place before you are able to issue any type-2 command.

To learn more about setting up the OM and the overall CSL, refer to:

- CSL definition and tailoring information in *IMS Version 11 System Definition,* GC19-2444
- CSL administration information in *IMS Version 11 System Administration,* SC19-2443

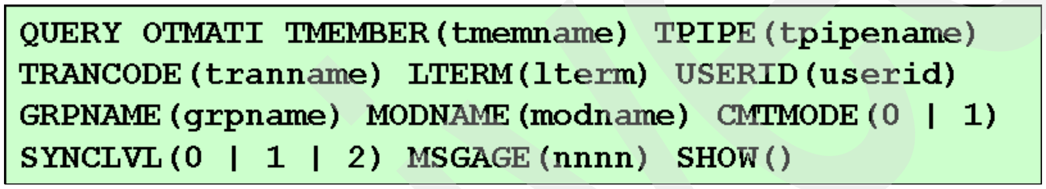
We review the following commands in more detail:

- QUERY OTMATI command
- Dynamic OTMADESC commands

4.4.1 QUERY OTMATI command

When a message is sent through OTMA, either to an IMS system or to a client, it is represented by a *transaction instance block* (TIB). The QUERY OTMATI command retrieves information related to these TIBs, to give you information about your OTMA environment, including current OTMA members, clients, and message volume. A TIB can represent different kinds of messages, such as a CM0 (commit-then-send) or CM1 (send-then-commit) input message that has not yet been enqueued. It can also represent a CM1 message that is either enqueued but not executing, executing after being enqueued, or waiting for a client ACK. In addition, a TIB can represent an IMS conversation waiting for the next input message, or an orphaned transaction that occurs when IMS is unable to free a TIB in an error scenario. You can issue the QUERY OTMATI command on an active IMS, an XRF alternate, or RSR tracker system, with certain parameters to pinpoint the information you have in the command response.

Display information about an OTMA member's environment by issuing the QUERY OTMATI command with the syntax shown in Figure 4-11.



```
QUERY OTMATI TMEMBER(tmemname) TPIPE(tpipename)
TRANCODE(tranname) LTERM(lterm) USERID(userid)
GRPNAME(grpname) MODNAME(modname) CMTMODE(0 | 1)
SYNCLVL(0 | 1 | 2) MSGAGE(nnnn) SHOW()
```

Figure 4-11 Command syntax for querying an OTMA member's environment

Table 4-3 lists the command parameters. For further information, refer to *IMS Version 11 Commands, Volume 2: IMS Commands N-V,* SC19-243.

Table 4-3 QUERY OTMATI parameters

Parameter	**Meaning**
TMEMBER	Specifies an OTMA transaction member name (1 - 16 characters). The member is a client of OTMA, such as IMS Connect.
TPIPE	Specifies an OTMA transaction pipe name (1 - 8 characters).
TRANCODE	Specifies a transaction code (1 - 8 -characters) associated with the program that is scheduled.
LTERM	Specifies an override LTERM name (1 - 8 characters) that is included in the OTMA message state prefix. Any LTERM name that exceeds eight characters is flagged as an error.
USERID	Specifies a RACF user ID (1 - 8 characters) that is included in the security prefix of the message.
GRPNAME	Specifies a RACF Group ID (1 - 8 characters) that is included in the security prefix of the message. Any group name that exceeds eight characters is flagged as an error.
MODNAME	Specifies an output descriptor name(1 - 8 characters) that is included in the OTMA message state prefix. This descriptor is associated with the transaction or the program to be scheduled.
CMTMODE	Specifies the commit mode of the workload to be displayed. Specify 0 to display all workloads that are in commit-then-send (CM0) mode, which is supported on both persistent and transaction sockets and supports only synchronization level CONFIRM. Specify 1 to display all workloads that are in send-then-commit (CM1) mode, which is supported on both persistent and transaction sockets and supports synchronization level NONE, CONFIRM, and SYNCH.
SYNCLVL	Specifies the synchronization level. Specify 0 for a synchronization level of NONE, which requires no acknowledgment from the client. Specify 1 for a synchronization level of CONFIRM, which requires the client to acknowledge delivery of output messages. Specify 2 for a synchronization level of SYNCH, for two-phase commit processing that involves multiple participants in syncpoint processing that is managed through RRS.
MSGAGE	Specifies the minimum amount of clock time since the message (YTIB) became active. The MSGAGE value is in seconds and must be an integer.
SHOW	Specifies the output fields to be returned. This parameter uses specific output field filters that you can have included in the output. Table 4-4 lists the filters used with the SHOW parameter.

Table 4-4 describes the SHOW parameter filters. (See the SHOW parameter, listed in Table 4-3 on page 129.)

Table 4-4 SHOW output field filters for the QUERY OTMATI command

Filter	**Meaning**
AGINGVAL	Displays the aging value (how often the cached user ID ACEE should be refreshed). The aging value is either the message aging value or the client aging value, and is from the message control prefix or from the message state prefix, respectively.
ALL	Displays all information about the OTMA message workload.
CMTMODE	Displays the commit mode. ► 0 represents commit-then-send mode. ► 1 represents send-then-commit mode.
CTTKN	Displays the context token when a transaction is in a two-phase commit that involves multiple participants in syncpoint processing managed through RRS.
GRPNAME	Displays the RACF group ID.
LTERM	Displays the override LTERM name.
MODNAME	Displays the override MODNAME.
MSGAGE	Displays the minimum age in seconds since the message (TIB) became active.
MSGCNT	Displays the total number of active TIBs associated with messages, depending on the various parameter values requested.
MSGTKN	Displays the correlator token.
TIMEOUTVAL	Displays the timeout value for CM1 that is missing ACK.
TRANCODE	Displays the transaction code that is associated with the message.
TYPE	Displays the message type up to four of the following types: ► SMB: SMB transaction ► CPC: CPIC transaction ► CMD: IMS command ► APC: Message switch ► RCV: Recoverable transaction ► CON: Conversational transaction ► EMH: Fast Path transaction ► RSP: Transaction response
USERID	Displays the user ID that is included in the security data prefix of the message.
SYNCLVL	Displays the synchronization level of NONE, CONFIRM, or SYNCH.

One useful way to use the QUERY OTMATI command is with the SHOW(TRANCODE) filter, which is abbreviated in SHOW(TRAN). This filter enables you to see each specific transaction and its associated message quantity. If you see a high message volume for one particular transaction, the filter enables you to take preventative action before a flood condition arises. See Example 4-22 on page 131.

Example 4-22 Output of QUERY OTMATI command with the SHOW(TRAN) parameter

```
Response for: QUERY OTMATI MSGAGE(5) SHOW(TRAN)                             More: >
MbrName   Tmember     TpipeName      CC    CCText
IMSA      MQ          CSQ81234        0    Complete successfully
IMSA      MQ          CSQ81234        0    Complete successfully
IMSB      ICONN1      APPLB           0    Complete successfully
IMSB      ICONN1      APPLB           0    Complete successfully
IMSB      WAS         APPLC           0    Complete successfully

(Page to the right within the TSO SPOC application to see a continuation of the the output)

Response for: QUERY OTMATI MSGAGE(5) SHOW(TRAN)                     More: <
MsgCnt    MessageAge      Transaction
     2             5     ACCTINQ
     1             7     NEWACCT
     2             8     INVQRY
     9             6     CHKSTAT
     4             9     SRCHPART
```

4.4.2 Dynamic OTMADESC commands

IMS 11 introduces four OTMA commands, which enable you to dynamically manage your OTMA descriptor resources, which are used to route IMS application callout requests to specific TPIPE destinations. Without an OTMA descriptor, you have to code the assembler-routing exits in order to pass callout-message requests to an external java application through the IMS TM Resource Adapter. Prior to IMS 11, restarting IMS was required before OTMA descriptor changes could take effect, resulting in an outage. For more information about OTMA descriptors, refer to *IMS Version 11 Communications and Connections,* SC19-2433.

The following type-2 commands allow dynamic modification to OTMA descriptor destinations without requiring the IMS system to be restarted:

- CREATE OTMADESC
- UPDATE OTMADESC
- DELETE OTMADESC
- QUERY OTMADESC

When you use these commands, the destination, to which an IMS application-callout request is issued, is affected. A x'221B' log record is written to the IMS log when the CREATE, UPDATE, or DELETE command is issued, and a '4035' log record is written for the descriptors at checkpoint time. We now describe each of these commands.

Creating OTMA descriptors

Dynamically create an OTMA descriptor by issuing the CREATE OTMADESC command with the syntax shown in Figure 4-12.

```
CREATE OTMADESC NAME(destname*)
SET(TYPE(IMSCON | NONOTMA)
TMEMBER(tmem) TPIPE(tpipename) SMEM(N | Y)
SYNTIMER(syntimer) ADAPTER(adaptername)
CONVRTR(convertrname))
```

Figure 4-12 Syntax to create an OTMA descriptor with the CREATE OTMADESC command

When you create an OTMA descriptor with this command, it is added to the active IMS system. The statically created descriptors that you defined within the DFSYDTx PROCLIB member still remain in the system. The actual content of the descriptors do not change with this enhancement, so the parameters (required and optional) remain the same.

Table 4-5 summarizes the meaning of each parameter in order of appearance, and indicates whether it is required or optional. For additional detail, refer to the *IMS Version 11 Commands, Volume 1: IMS Commands A-M,* SC19-2430.

Table 4-5 CREATE OTMADESC command parameter definitions

Parameter	Meaning
NAME(destname*)	A required parameter that specifies a destination name (1 - 8 characters). The destination name can have an asterisk (*) at the end to mask groups of names. An asterisk by itself is a valid name and is an entry in the table of destination-routing descriptors. A masked name that encompasses another masked name does not have to be created in the order from most specific to most generic. However, the search-order starts from the most specific to the most generic. Creating a destination name that already exists in the system is an error.
TYPE(<u>IMSCON</u> | NONOTMA)	An optional parameter that can either be IMSCON or NONOTMA. IMSCON takes the default value if TYPE parameter is not specified. If TYPE(IMSCON) is specified, TMEMBER is a required parameter and the remaining parameters (such as TPIPE, SMEM, or ADAPTER) are optional. If TYPE(NONOTMA) is specified, coding any of these required or optional parameters is not allowed.
TMEMBER()	Required only for TYPE(IMSCON), this parameter is an OTMA TMEMBER name (1 - 16 characters). Coding this parameter with TYPE(NONOTMA) causes an error. If SMEM(Y) is specified, the value is a maximum of 4-character super member name.
TPIPE()	An optional parameter that is a TPIPE name (1 - 8 characters) when TYPE(IMSCON) is specified. If this parameter is not coded, the TPIPE name is the destination name specified in the NAME parameter. This parameter is not valid if TYPE(NONOTMA) is specified. If the TPIPE is not coded and the NAME is a mask, the TPIPE will be a specific destination name that matched the mask in the table of destination routing descriptors and not the NAME parameter.
SMEM(N | Y)	An optional parameter that can either be N (no) or Y (yes) to indicate whether the TMEMBER name specified in the TMEMBER parameter is a super member. If the TMEMBER name is a super member, the length of the TMEMBER name has a maximum of four characters. This is an optional parameter when TYPE(IMSCON) is specified. It is not a valid parameter and can cause an error if TYPE(NONOTMA) is specified.
SYNTIMER	An optional parameter that specifies the timeout value that when reached, the synchronous callout reply expires because an ACK/NACK or reply from the client has not been received. The value, which is expressed in hundredths of a second, must be numeric and within a range of 0 through 999999, inclusive. When the parameter is coded, a numeric value must be specified. If SYNTIMER(0) is specified, the value defaults to 1000, which is equivalent to 10 seconds.
ADAPTER	An optional parameter for TYPE(IMSCON). The value is a name (1 - 8 characters) that identifies the IMS Connect adapter. If TYPE(NONOTMA) is specified, you cannot specify this parameter.
CONVRTR	A required parameter only if the ADAPTER parameter is specified. The value is a converter name (1 - 8 characters) used by the adapter. Coding this parameter without the ADAPTER parameter or with TYPE(NONOTMA) can cause an error.

You can create OTMA descriptors either individually or to mask groups with an asterisk (*) at the end of the NAME parameter value. See Example 4-23.

Example 4-23 Creating OTMA descriptors with CREATE OTMADESC: individual and by group

```
Response for: CREATE OTMADESC NAME(OTMACL99) TYPE(IMSCON) TMEMBER(HWS1)
TPIPE(HWS1TP01)
DestName   MbrName     CC
OTMACL99   IMSA         0

Response for: CREATE OTMADESC NAME(OTMACL*) TYPE(IMSCON) TMEMBER(HWS2)
DestName   MbrName    CC
OTMACL*    IMSA        0
```

Updating OTMA descriptors

Dynamically update an OTMA descriptor without having to restart your IMS system by using an UPDATE OTMADESC command with the syntax shown in Figure 4-13.

```
UDPATE OTMADESC NAME(destname*)
SET(TYPE(IMSCON | NONOTMA) ADAPTER(adaptername)
CONVRTR(convertrname) SMEM(N | Y)
SYNTIMER(syntimer) TMEMBER(tmem)
```

Figure 4-13 Command syntax for dynamically updating an OTMA descriptor

You can use this command to update OTMA descriptors that were created statically in the DFSYDTx PROCLIB member or created dynamically with a CREATE OTMADESC command. Special consideration should be given when changing the TYPE parameter with this command. If an OTMA descriptor is being changed from TYPE(NONOTMA) to TYPE(IMSCON), you must include the TMEMBER parameter when issuing the UPDATE command. When updating an OTMA descriptor of TYPE(IMSCON), you can specify the other parameters in the command with blanks, for example `TPIPE()`, to erase any pre-existing value for that parameter. When changing an OTMA descriptor from TYPE(IMSCON) to TYPE(NONOTMA), keep in mind that all parameter values for the descriptor are automatically deleted.

Table 4-6 summarizes the meaning of each parameter in order of appearance and indicates whether it is required or optional. For additional detail, refer to *IMS Version 11 Commands, Volume 2: IMS Commands N-V,* SC19-2431.

Table 4-6 UPDATE OTMADESC command parameter definitions

Parameter	Meaning
NAME(destname*)	A required parameter whose value cannot be modified.
SET()	Specifies to update the descriptors with at least one of the following parameters. The update applies to each of the descriptor names specified in the NAME parameter. If an asterisk is appended to the descriptor name, the update is not applied to the group of names it is trying to mask.
ADAPTER()	An optional parameter for TYPE(IMSCON) that identifies the IMS Connect adapter name (1 - 8 characters). To clear the value of this parameter, specify the parameter with no value, for example, ADAPTER(). If the ADAPTER parameter is deleted, the CONVERTER parameter is also deleted. ADAPTER and TYPE(NONOTMA) are mutually exclusive.

Parameter	Meaning
CONVERTR()	A required parameter only if the ADAPTER parameter is specified. This parameter is a converter name (1 - 8 characters) used by the adapter. The value of this parameter is automatically deleted if the value of the ADAPTER parameter is being deleted. Coding this parameter without the ADAPTER parameter or with TYPE(NONOTMA) is not valid.
SMEM(N I Y)	An optional parameter that can have a value of either No or Yes to indicate whether the TMEMBER name specified in the TMEMBER parameter is a super member. To clear the value of this parameter, specify the parameter with no value, for example, SMEM(). Deleting SMEM() defaults to SMEM(N). If the TMEMBER name is a super member, the length of the TMEMBER name has a maximum of 4 characters. This is an optional parameter when TYPE(IMSCON) is specified. SMEM and TYPE(NONOTMA) are mutually exclusive.
SYNTIMER(syntimer)	An optional parameter used for synchronous callout processing. In synchronous callout processing, a request is canceled if an ACK/NACK or response is not received by OTMA from the client within the time specified. This value is expressed in hundredths of a second or milliseconds. The value can be numeric with a maximum length of six digits. The value has a range of 0 - 999999; if zero is specified, it defaults to 10 seconds or 1000 hundredths of a second. If you specify SYNTIMER() or SYNTIMER(), the timeout value will be reset as if no timeout value has been specified.
TMEMBER()	A required parameter for TYPE(IMSCON). If the TYPE of the descriptor is being changed from IMSCON to NONOTMA, the value of this parameter is deleted. If the TYPE of the descriptor is being changed from NONOTMA to IMSCON, then the TMEMBER parameter must be coded. The value is a 16-character OTMA TMEMBER name or a 4-character super member. TMEMBER and TYPE(NONOTMA) are mutually exclusive.
TPIPE()	An optional parameter that is a 1- to 8-character TPIPE name when TYPE(IMSCON) is specified. To delete the value of this parameter, specify the parameter with no value, for example, TPIPE(). If this parameter is not specified, the TPIPE name is the destination name specified in the NAME parameter. TPIPE and TYPE(NONOTMA) are mutually exclusive.
TYPE(IMSCON I NONOTMA)	A required parameter that can either be IMSCON or NONOTMA. This parameter determines whether the output is destined for IMS Connect (IMSCON) or non-OTMA (NONOTMA). If TYPE(IMSCON) is specified, the TMEMBER is a required parameter and the rest of the parameters, such as TPIPE, SMEM, and ADAPTER, are optional. If TYPE(NONOTMA) is specified, coding any of these optional parameters is not allowed. The destination routing descriptors can be changed from TYPE(IMSCON) to TYPE(NONOTMA) or vice versa. If the TYPE parameter is changed from IMSCON to NONOTMA, the values of the remaining parameters (TMEMBER, TPIPE, SMEM, ADAPTER, and CONVERTR) are deleted. If the TYPE parameter is changed from NONOTMA to IMSCON, the TMEMBER parameter must be coded. For TYPE(IMSCON), the optional parameters can be coded with no values that indicate whether to delete any values parameter for the parameters in question; for example, coding TPIPE() clears the value of TPIPE(PIPERONE) for the descriptor.

Example 4-24 shows output from updating an OTMA descriptor with the UPDATE OTMADESC command.

Example 4-24 Updating an OTMA descriptor using the UPDATE OTMADESC command

```
Response for: UPDATE OTMADESC NAME(OTMACL*,OTMACL99) SET(TPIPE(HWS1TP02))
DestName   MbrName     CC
OTMACL99   IMSA         0
OTMACL*    IMSA         0
```

Deleting OTMA descriptors

Delete an OTMA descriptor by issuing the DELETE OTMADESC with the syntax in Figure 4-14.

```
DELETE OTMADESC NAME(destname*)
OPTION(NOWILDCARD | WILDCARD)
```

Figure 4-14 Command syntax for dynamically deleting an OTMA descriptor

You are able to delete OTMA descriptors individually or by group, using an asterisk (*). The two parameters for this command are listed in Table 4-7. For more details about this command, refer to *IMS Version 11 Commands, Volume 1: IMS Commands A-M,* SC19-2430.

Table 4-7 DELETE OTMADESC command parameter definitions

Parameter	**Meaning**
NAME(descname*)	A required parameter that specifies a destination routing descriptor name (1 - 8 characters). The destination routing descriptor name can have an asterisk (*) at the end to mask a group of names. The asterisk can be used by itself to mask all defined descriptor names in the system. Issuing the DELETE command with a name of asterisk only deletes all entries in the destination routing descriptors. If the user intends to delete all entries in the table, DELETE NAME(*) OPTION(WILDCARD) must be specified. All entries including the entry of an asterisk (*) are deleted.
OPTION(NOWILDCARD | WILDCARD)	An optional parameter, specify NOWILDCARD (default) to delete the descriptor having a name with an asterisk as an entry from the table of destination routing descriptors. Specify WILDCARD to delete the groups of names that the asterisk is masking. If the name with the asterisk is also an entry in the table, the name with the asterisk itself is deleted.

To delete a group of OTMA descriptors, make sure that you include an asterisk in the NAME parameter value and specify OPTION(WILDCARD) when issuing the command. If no OPTION is specified, the default of OPTION(NOWILDCARD) is taken and only the descriptor name containing the asterisk will be deleted; the other descriptors masked by it remain in the table of destination routing descriptors. In other words, the masking descriptor is treated as a standalone entry in the table.

Let us now review several examples. If we issue a QUERY OTMADESC command to reveal all descriptors that a certain masking descriptor name represents, we see how the DELETE OTAMDESC command shows different output depending on whether we specify the OPTION parameter. Example 4-25 shows information displayed for a descriptor named OTMACL*. We can see that it is masking one other descriptor named OTMACL99, which is visible because OPTION(WILDCARD) was included.

Example 4-25 Querying an OTMA descriptor using a masked NAME() value

```
Response for: QUERY OTMADESC NAME(OTMACL*) OPTION(WILDCARD)
DestName    MbrName  CC Type    TMember  TPipe        SMem
OTMACL*     IMSA      0 IMSCON  HWS2                  N
OTMACL99    IMSA      0 IMSCON  HWS1     HWS1TP01     N
```

Now we delete several descriptors using this same NAME() parameter. Example 4-26 on page 136 shows output for the DELETE OTMADESC command when an asterisk (*) is included in the NAME() parameter value, but with no OPTION specified. In this case, the only descriptor deleted is `OTMACL*` but the OTMACL99 descriptor entry remains in the destination routing descriptor table.

Example 4-26   Deleting OTMA descriptor by DELETE OTMADESC command: no OPTION

```
Response for:DELETE OTMADESC NAME(OTMACL*)
DestName  MbrName     CC
OTMACL*   IMSA         0
```

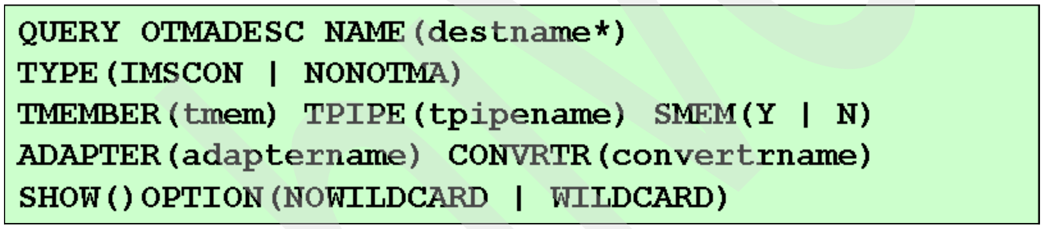
Example 4-27 shows output for the DELETE OTMADESC command when an asterisk (*) is included in the NAME() parameter value, but with OPTION(WILDCARD) specified this time. We see that the OTMACL99 descriptor is now also deleted because of the inclusion of this parameter.

Example 4-27   Deleting OTMA descriptor by DELETE OTMADESC command: OPTION

```
Response for:DELETE OTMADESC NAME(OTMACL*) OPTION(WILDCARD)
DestName   MbrName     CC
OTMACL99   IMSA         0
OTMACL*    IMSA         0
```

Querying OTMA descriptors

Display the attribute values of an OTMA descriptor by issuing the QUERY OTMADESC with the syntax shown in Figure 4-15.



```
QUERY OTMADESC NAME(destname*)
TYPE(IMSCON | NONOTMA)
TMEMBER(tmem) TPIPE(tpipename) SMEM(Y | N)
ADAPTER(adaptername) CONVRTR(convertrname)
SHOW()OPTION(NOWILDCARD | WILDCARD)
```

Figure 4-15   Command syntax for querying an OTMA descriptor's attribute values

Table 4-8 summarizes the meaning of each parameter in order of appearance and indicates whether it is required or optional. For additional detail, refer to the *IMS Version 11 Commands, Volume 2: IMS Commands N-V,* SC19-2431.

Table 4-8   QUERY OTMADESC command parameter definitions

Parameter	Meaning
ADAPTER()	Specifies an adapter name (1 - 8 characters) that identifies the IMS Connect adapter. This parameter is optional for TYPE(IMSCON). The information is displayed only if it is filtered or coded in the SHOW parameter. This parameter is ignored if TYPE(NONOTMA) is specified.
CONVRTR()	Specifies a converter name (1 - 8 characters) that is associated with the adapter specified with the ADAPTER parameter. The information is displayed if it is filtered or included in the SHOW parameter.
NAME(destname*)	A required parameter that specifies a destination name (1 - 8 characters). The destination name can have an asterisk (*) by itself or appended to mask group of names. An asterisk by itself is an entry in the table of destination routing descriptors.
OPTION(NOWILDCARD | WILDCARD)	NOWILDCARD displays a name with an asterisk as an entry from the table of destination routing descriptors. WILDCARD displays the groups of names that are being masked by the name with an asterisk. If the name with the asterisk is also an entry in the table, the name itself is displayed.
SHOW()	Specifies which information to display in the output fields. This parameter uses specific output field filters that you can have included in the output. Table 4-9 lists the filters.

Parameter	Meaning
SMEM(Y | N)	An optional parameter that specifies whether the TMEMBER name is a super member. SMEM and TYPE(NONOTMA) are mutually exclusive.
TMEMBER()	An optional parameter that is used to filter the TMEMBER name. TMEMBER and TYPE(NONOTMA) are mutually exclusive.
TPIPE()	An optional parameter that is used to filter the TPIPE name. TPIPE and TYPE(NONOTMA) are mutually exclusive.
TYPE(IMSCON | NONOTMA)	An optional parameter that can be either IMSCON or NONOTMA. Both output types are displayed if the parameter is not specified. To filter the display, specify either IMSCON or NONOTMA. To include the other parameters, such as TMEMBER, TPIPE, SMEM, ADAPTER or CONVERTR, use the SHOW parameter. If TYPE(NONOTMA) is specified, however, the remaining display parameters show blanks even if the SHOW parameter is specified.

Table 4-9 describes the SHOW parameter filters. (See the SHOW parameter, listed in Table 4-8.)

Table 4-9 SHOW output field filters for the QUERY OTMADESC command

Filter	Meaning
ADAPTER	Displays the name that identifies the IMS Connect adapter.
ALL	Displays all subparameters.
CONVRTR	Displays the converter name used by the adapter.
SMEM	Displays the super member indicator.
SYNTIMER	Displays the timeout value for synchronous callout processing.
TMEMBER	Displays the name of TMEMBER or the Super Member if SMEM indicates as such.
TPIPE	Displays the TPIPE name under a TMEMBER.
TYPE	Displays either IMSCON or NONOTMA for the type of descriptor.

Example 4-25 on page 135 shows output from a QUERY OTMADESC command.

Recovering OTMA descriptor changes

You are able to recover dynamic OTMA descriptor changes across warm and emergency restarting of IMS, but not across cold-starting of IMS. During warm and emergency restarts, checkpoint records are read in order to reconstruct the destination routing descriptor table. The IMS log is also read during the emergency restarting of IMS. If you cold start IMS, dynamic changes you made to your OTMA descriptors in the previous IMS instance with type-2 commands are lost. To avoid losing these changes, you have to manually update the DFSYDTx PROCLIB member to reflect the dynamic changes you made during the prior execution of IMS.

IMS systems that support dynamic OTMA descriptor changes

Of the new OTMA type-2 descriptor commands introduced in IMS 11, only the QUERY OTMADESC command can be issued on systems other than the active IMS, including XRF alternate and RSR tracker systems. If you attempt to issue a CREATE, UPDATE, or DELETE OTMADESC command on these types of alternate systems, it is rejected. However, if either of these systems becomes the active IMS system, you may then issue these commands.

The CREATE, UPDATE, and DELETE OTMADESC commands are also invalid when they are issued in an FDBR region.

XRF alternate and RSR tracker systems are kept updated with dynamic descriptor changes by way of checkpoint and log records. Keep in mind that if you issue a QUERY OTMADESC command on an XRF alternate or RSR tracker system, the output might reflect information that is different than that of the active system because the checkpoint and log records might not have had time to be percolated to these other systems' destination routing descriptor tables yet.

Migration

In IMS 11, OTMA descriptors defined in the DFSYDTx PROCLIB member are not required to be in order of most specific to most generic, as they were in IMS 10. For more information, refer to Chapter 7, "Installation and migration considerations" on page 227.

4.5 Transaction expiration

IMS 11 introduces a capability that allows a transaction message to expire after a certain amount of time has elapsed. This capability can be useful when either IMS or the network is experiencing a delay, in turn causing the client to time out and no longer requiring a response.

An IMS 11 transaction parameter, EXPRTIME, allows you to specify a value in hundredths of seconds that IMS can compare to a message's age before processing the message. This comparison occurs when a message is scheduled in a dependent region and an application program issues a GU call to the I/O PCB to obtain a message segment. If the transaction's EXPRTIME value is less than the time that has elapsed since the message reached the IMS system, the message is discarded and not processed. In this section, we discuss how to set the timeout value, what happens when a transaction expires, and why this enhancement is beneficial.

Restriction: Certain types of transactions are only checked for expiration if they have OTMA message-level specification capability. Messages for the following transaction types are not checked for expiration during the GU phase (only during receiving and enqueuing phases):

- MSC remote
- IFP
- IMS conversational
- Switched-to for program-to-program switches

4.5.1 Setting the EXPRTIME value

You can set the EXPRTIME value in a variety of ways:

- The TRANSACT macro within System Definition (SYSGEN)
- The Destination Creation user exit routine (DFSINSX0)
- A CREATE or UPDATE dynamic resource definition command

In this section, we explore each of these options.

TRANSACT macro

Specify the EXPRTIME parameter value in hundredths of seconds using the range 0 - 65535 on the TRANSACT macro statements. The default value is 0, which means that no timeout value is set and the transaction is not capable of expiring.

If you specify a value that is outside the 0 - 65535 range, you receive the following warning message:

```
G316 EXPRTIME OPERAND INVALID, DEFAULT OF ZERO ASSUMED
```

In this instance, the default value of 0 is taken for the parameter setting. Otherwise, the value is used in the comparison to the age of a message before IMS attempts to process it.

Destination Creation user exit routine (DFSINSX0)

The purpose of the Destination Creation user exit is to dynamically create a transaction or LTERM destination when a message arrives with an unknown destination. Using the exit, you specify parameter values for the resource it dynamically creates. Support for the EXPRTIME transaction attribute has been added to the user exit. When you set this attribute by using the exit, it applies to all dynamically created transactions. In a shared queues environment, it applies also to include transactions that are scheduled on the local IMS system and in a shared queues environment on a back-end IMS system.

Dynamic resource definition

You are able to dynamically create, update, and query transactions using dynamic resource definition (DRD) commands using the CREATE, UPDATE, and QUERY commands respectively. Support for the EXPRTIME parameter has been added to each command, as shown in Figure 4-16, where `tranname` is the name of the transaction and `sssss` is hundredths of seconds.

```
CREATE | UPDATE | QUERY TRAN NAME(tranname)
SET(EXPRTIME(sssss))
```

Figure 4-16 DRD command syntax for using the EXPRTIME parameter

Type-2 commands issued to IMS are written to the IMS log as x'22' log records, including the DRD commands just mentioned. This means that if IMS abends and you have to warm-start or emergency-restart it, the log will be read during the restart and the EXPRTIME parameter value previously set is recovered. In addition, if you are using an XRF alternate IMS system, it too is aware of this transaction parameter setting, because activity that occurs on the active IMS system is propagated to the alternate system through x'22' log records.

Resource Definition Data Set support

When you use DRD, transaction resource information (and other MODBLKS data) is stored in the Resource Definition Data Set (RDDS). This data set now recognizes the EXPRTIME transaction parameter value, which is included in RDDS I/O activities like DRD import and DRD export. These I/O activities commonly occur automatically during the restart, system checkpoint, or initiated by a type-2 IMPORT or EXPORT command.

To learn more about how DRD automatically imports and exports data to and from the RDDS, refer to the DRD information in *IMS Version 11 System Definition,* GC19-2444.

For details about the IMPORT and EXPORT commands, refer to *IMS Version 11 Commands, Volume 1: IMS Commands A-M,* SC19-2430.

The RDDS Extraction Utility allows you to read the contents of the RDDS and create transaction-related SYSGEN macro statements or type-2 CREATE commands. This utility has been enhanced to recognize the new EXPRTIME transaction parameter and include it in the output that it generates.

Applicable scope of expiration time

The expiration value that you specify for the EXPRTIME parameter applies to incoming messages for all IMS transactions. However, when using OTMA, you may override this setting by specifying a value in the OTMA message prefix. In this case, the setting would apply to individual messages rather than all messages that are queued to a particular transaction. Refer to 4.3.2, “OTMA transaction expiration and input message timeout” on page 115 for more information about specifying this option for individual messages in the OTMA message prefix.

4.5.2 Actions that occur when a transaction expires

When IMS detects an expired transaction, it writes a x’67D0’ log record, issues messages, and depending on the input message source, might also pseudoabend.

Non-OTMA transaction message expiration

When a non-OTMA transaction expires, IMS issues message DFS3688I to the terminal, indicating the expired transaction name, the EXPRTIME value, and the age of the message.

OTMA transaction message expiration

When an OTMA transaction message expires, an abend 0243 is issued in addition to either a DFS555I or DFS2224I message, which indicates the expired transaction name, the associated IMS system, and actual input message data. You can find details about the expiration process in “OTMA transaction expiration process” on page 116.

For more information about messages DFS3688I, DFS555I, and DFS2224I, refer to *IMS Version 11 Messages and Codes, Volume 1: DFS Messages,* GC18-9712.

For details about abend code 0243, refer to *IMS Version 11 Messages and Codes, Volume 3: IMS Abend Codes,* GC18-9714.

4.5.3 Enhancement advantages

When IMS processes a transaction and the client no longer requires an output response, unnecessary processor cycles are used, incurring processing costs. With the transaction expiration enhancement, processing efficiency is increased because IMS 11 has a mechanism to determine whether processing a transaction is necessary. Because of this increased efficiency, IMS applications now wait a shorter period of time to access the resources they require when other transactions are entered.

4.6 Multiple Systems Coupling enhancements

Multiple Systems Coupling (MSC) allows IMS systems that are in separate locations to communicate with each another through links that have been established between them. In this environment, messages can be received on one IMS and routed to another remote IMS. Message routing is handled by the TM and MSC Routing and Control user exit known as DFSMSCE0.

IMS 11 introduces two enhancements to the DFSMSCE0 user exit:

- The capability of routing messages with affinity in a shared queues IMSplex
- An expansion of your ability to create, add to, or modify accounting data in the user prefix within a message from an IMS application before it is sent back to the originating source

In this section, we explore each of these enhancements.

4.6.1 Shared queues affinity routing

In IMS 11, you are able to use the DFSMSCE0 user exit to enforce message affinity routing within a shared queues IMSplex. Previously, the only available method for controlling transaction processing was by using region and class scheduling to this effect, which was not always straightforward. This enhancement increases scheduling efficiency by eliminating the need for multiple regions to make a bid for a transaction when in reality, only one of them is capable of processing it. Affinity processing ensures that the transaction is only attempted to be processed on one IMS system.

When MSC is being used in a shared queues IMSplex, it is common that some IMS systems within the IMSplex are treated as *back-end* (BE) systems that process incoming messages. The messages are initially received by *front-end* (FE) IMS systems that reside on another LPAR, which send them to the BE systems through the shared queues. Message traffic between FE and BE systems in a SQ IMSplex is through the shared queues, not MSC links. MSC links between IMSs in the same SQ IMSplex, (which might still exist because they have not yet been removed after a migration from MSC) will receive error message

```
DFS2149 PARTNER IMS IN SAME SHARED QUEUES GROUP - RESTART ABORTED LINK xxx
```

Let us explore this scenario further.
When the DFSMSCE0 user exit is used to send these messages to the BE systems through the shared queues, it specifies the name of the shared queue (SQNAME) that the messages sit on while waiting to be processed. Because this is a shared queue, any of the BE IMS systems can process the message. But with this enhancement, the exit can now append the preferred IMSID of the system to the SQNAME within the message prefix to route the message with affinity to that specific IMS system. Appending the IMSID to the SQNAME is the key that establishes affinity between the transaction message and the IMSID of the system. This is only half of the requirements for affinity processing, however.

Before a transaction can be processed with affinity on the designated IMS system, you are required to first start the transaction with affinity.

Registering transactions for affinity

To register a transaction as having affinity to an IMS system, you can enter the following type-2 command on the system:

```
UPDATE TRAN NAME(<trancode>) START(SCHD) OPTION(AFFINITY)
```

An example is shown in Example 4-28.

Example 4-28 Registering APOL12 transaction with affinity on IMS1 with UPDATE command

```
Response for: UPDATE TRAN NAME(APOL12) START(SCHD) OPTION(AFFIN)
Trancode MbrName    CC
APOL12   IMS1        0
```

You may also enter the following type-1 command on the system to register the transaction with affinity to the IMS system:

```
/START TRAN (<trancode>) AFFINITY
```

See Example 4-29.

Example 4-29 Registering APOL12 transaction with affinity on IMS1 with /START command

```
*277 DFS996I *IMS READY*
 R 277,/START TRAN APOL12 AFFINITY
 DFS058I START COMMAND COMPLETED
 DFS000I     *09202/143650*
*278 DFS996I *IMS READY*
```

If you have to restart an IMS system after a transaction has already registered for affinity with it, the registration is recovered only across a warm start. In the event of a cold start, you would have to re-register the transaction for affinity with the IMS system. This would be an area where tooling could come in handy. After you have registered transactions for affinity to a particular IMS system (or systems), this registration remains until the next IMS cold start. Currently, no way exists to unregister transactions for affinity.

Two pieces required to process messages

We see here, that two pieces are required to process messages with affinity in a shared queues IMSplex environment:

- The DFSMSCE0 exit has to establish affinity between a transaction and an IMS system by appending the desired IMSID to the destination shared queues name.
- The transaction has to be registered with affinity on the IMS system by a command.

Normal versus affinity transaction registration

Let us explore the scenarios where one of these pieces (mentioned in the previous section) is missing. Let us say that you try to use the DFSMSCE0 user exit to establish affinity between a transaction and an IMS system, but do not register the transaction for affinity on the system with a command. How would incoming messages for this transaction be handled in this case? These messages would go through normal registration (versus affinity registration) and would remain on the shared queue until the appropriate command is entered with the AFFINITY option included, thereby registering the transaction with affinity to the system. This scenario might occur if you attempt to register a transaction with affinity to an IMS system using a command, but forgot to include the AFFINITY parameter.

Conversely, what would happen if you did register a transaction with affinity to an IMS system with a command, but did not set up the DFSMSCE0 user exit to process the transaction messages with affinity? The message would remain on the shared queue until the transaction was registered normally (for example, without affinity) on an IMS system. At that point the system would process this message. Therefore, as you can see, a transaction can be registered differently among IMS systems that reside within the same IMSplex.

Querying transactions for affinity

If you are unsure of the transactions that you have already registered to specific IMS systems with affinity, you can issue one of the following commands to determine this information:

- QUERY TRAN NAME(ALL) SHOW(AFFIN)
- QUERY TRAN NAME(ALL) SHOW(STATUS)
- /DIS TRAN ALL QCNT

The command output shows which transactions have been registered with affinity.

Enhancement advantages

One way to exploit the DFSMSCE0 user exit is to use it for workload balancing among the systems in the IMSplex. You can write the exit to balance the work among the different message destinations in the IMSplex, and in turn register all transactions with affinity. The exit routine will then always be called and the messages will be processed in an even distribution across the IMSplex according to the exit routine logic.

Several restrictions are associated with affinity routing if APPC/OTMA is being used. When using the TM and MSC Message Routing and Control user exit routine to route with affinity, these restrictions are overridden.

Routing with affinity mirrors a local-queues scenario because only one IMS system is being used to process the message. When migrating to an IMSplex using a shared-queues environment, MSC users see the similarity because only one IMS system continues to process the message on the desired IMS system (if routing is done with affinity).

Migration considerations

If you are currently using the DFSMSCE0 user exit routine, you have to reassemble it after migrating to IMS 11. For details regarding migration, refer to Chapter 7, “Installation and migration considerations” on page 227.

4.6.2 DFSMSCE0 entry point

As you may know, the DFSMSCE0 user exit receives control when IMS receives a message from a terminal, through an MSC link, or when an application program issues a CHNG or ISRT call to insert a message. Each of these entry points within the user exit can change the transaction code or LTERM that the message is destined for, reject the message entirely, or designate that the message be processed on the local or a remote IMS system. Another capability of the exit is add to, modify, or create a user prefix within a message, which is what we discuss here.

IMS 11 expands your ability to create, add to, or modify information in the user prefix within a message from an IMS application before it is sent back to the originating source. Prior to IMS 11, the only opportunity the DFSMSCE0 user exit had to manipulate user prefix data was when an application program issued an ISRT to the IOPCB (or alternate PCB) or a CHNG call to a modifiable PCB. IMS 11 includes an entry point to DFSMSCE0, which allows it to gain control when an application program issues a GU to the IOPCB for the input message. At this point, the exit can modify the information in the user prefix, or add new information to it, such as accounting data. As with the entry points that are called during an application program’s ISRT or CHNG call, this new entry point called during a GU can also create a user prefix if one does not already exist in the message.

The benefit of this enhancement is that installations can have more options for including additional accounting data in the message before it is sent back to the originating source. Any information that is added to the prefix, becomes part of the message and is logged as such. Therefore, it can be used by tools for analysis.

Attention: You cannot use this new entry point to route messages to any IMS system, its main function is to allow you to add information to a message’s user prefix.

IMS provides a sample DFSMSCE0 user exit routine which demonstrates how you can store information in the user prefix. The sample is shipped in the `IMS.ADFSSMPL` library.

For more information about the DFSMSCE0 user exit routine, refer to *IMS Version 11 Exit Routines,* SC19-2437.

This function is retrofitted to IMS 10 by PTF; refer to A.1.3.2, “Compatibility PTFs for IMS 10” on page 308.

4.7 Full function response mode recovery

IMS 11 introduces the ability to recover full function response mode across terminal logoff/logon and user signoff/signon. With IMS 11, when you enter a response mode transaction and lose your connection to IMS before receiving the output response, the first message you can see upon reconnecting is the output response to your prior input. As a review, *response mode* refers an IMS system’s ability to accept an input transaction only if the output response for the previous transaction has been sent.

4.7.1 Advantages

Before IMS 11, the first messages you would see upon reconnecting were other queued messages, because the response to your previous input was treated as an asynchronous message after the IMS connection was broken. Or, if you entered another response mode transaction after reconnecting, you would receive the response for that transaction first, then any messages that were queued, and finally, the response to your original input transaction that you had entered before losing your previous IMS connection. By the time the response to your original response mode transaction arrived, it was unclear why the message was being sent, and it was not evident that it was connected to your previous input.

The ability of IMS 11 to recover response mode across terminal logoff/logon and user signoff/signon eliminates this confusion, ensuring that the response for a transaction you entered before losing a connection to IMS is delivered immediately after reconnecting. Furthermore, it ensures that no other transactions can be entered until you receive the response to the original transaction, maintaining the integrity of the original response mode status that existed in your previous IMS connection. Note that this enhancement does not allow response mode to be recovered across IMS restart.

4.7.2 Enablement

To activate this enhancement, specify RCVYRESP=YES in your IMS system’s DFSDCxxx PROCLIB member. Any nodes or users that you would like this setting to take effect for must also be enabled with local status recovery mode. In other words, these resources must either have SRMDEF=LOCAL specified in their owning IMS system’s DFSDCxxx PROCLIB member, or be set to local status recovery mode by a descriptor or exit routine.

As a review, resource status recovery mode is set by the SRMDEF= parameter for IMS resources that have *significant status*, or whose status is recovered across logoff, signoff, and IMS restart processes, and whose deletion is prevented in all of these instances. The SRMDEF= value is specified in an IMS system’s DFSDCxxx PROCLIB member and indicates that its resources, which have significant status, be stored either in log records or in a resource structure. The setting `SRMDEF=LOCAL` is the default for a local queues environment when no resource structure is present, whereas `SRM=GLOBAL` is the default for a shared queues environment where a resource structure is being used.

Restriction: Full function response mode recovery does not apply to global resources stored in a resource structure. If `SRMDEF=GLOBAL` is specified in an IMS system's DFSDCxxx PROCLIB member, or if sysplex terminal management is enabled (`STM=YES`), this capability cannot be used.

For more information about how a resource statuses are classified, refer to *IMS Version 11 Communications and Connections,* SC19-2433.

For details about the SRMDEF= parameter specification in the DFSDCxxx PROCLIB member, refer to *IMS Version 11 System Definition,* GC19-2444.

Retro-fit to IMS Versions 9 and 10

This enhancement is also available in previous versions of IMS as SPEs.

The APARs or PTFs are:

- IMS Version 9: PK53989/UK32266
- IMS Version 10: PK53423/UK32360

4.8 Shared queues scheduling enhancement

IMS 11 alleviates what is known as a *false schedule* in a shared queues environment. This condition can occur when a message from the shared queue is scheduled in a dependent region, but when an IMS application issues a Get Unique (GU) call, the message is no longer in the queue. False schedules unnecessarily consume system resources, increase costs and can be caused in different ways, as we now discuss.

The message that is no longer available to the IMS application might have been retrieved by another IMS system. Alternatively, the transaction could have been scheduled in another dependent region within the same IMS system because the parallel limit (PARLIM) count was reached in the original region. The PARLIM count (specified on the TRANSACT macro) indicates whether a transaction can be scheduled in parallel and if so, the maximum number of messages allowed for the schedule. When the PARLIM value has been reached, another processing region is scheduled in order to process any additional messages.

If you have a shared queues environment, you may have the PARLIM limit set to 0 or 1, which is a common practice. When a single message is enqueued locally to the front-end IMS system, the PARLIM limit is easily reached. This, in turn, can result in a false schedule. IMS 11 includes a shared queues enhancement that prevents an additional region from being scheduled when the PARLIM value is reached, unless there really is another message that has to be processed. This approach helps reduce the total number of false schedules in a shared queues environment where most messages are queued locally first, as previously mentioned.

4.9 Dynamic storage private buffer pool enhancement

IMS 11 adds an IMS Transaction Manager (TM) dynamic storage private buffer pool, DYNP, for module dynamic storage. This storage pool is used by OTMA and APPC for message processing.

IMS and DCC procedures are enhanced with the new DYNP parameter, which specifies a value for the upper expansion limit of the IMS TM dynamic storage private buffer pool. The value specified on the DYNP parameter can be in kilobytes, megabytes, or gigabytes.

The type-1 /DISPLAY POOL command is also enhanced with the new DYNP parameter to support the display how much storage is in use by IMS TM dynamic storage private buffer pool.

For more information about setting the DYNP parameter, refer to *IMS Version 11 System Definition,* GC19-2444.

5

DBRC enhancements

In this chapter, we discuss the following main enhancements to IMS Database Recovery Control facility:

- BPE-based DBRC helps you to more easily gather DBRC statistics and diagnostics, and a new DBRC REQUEST user exit is provided.
- Security override for non-production copies of RECON allows you to more easily manipulate copies of RECON.
- Removing old data from the RECONs allows you to remove unnecessary data from RECON that previously was tedious to remove.

We also provide information about DBRC migration to and coexistence with IMS 11.

5.1 BPE-based DBRC

In IMS 11, you can choose to run the DBRC address space under the Base Primitive Environment (BPE). This has several advantages:

- BPE allows multiple user exits of the same type, whereas standard DBRC does not.
- BPE allows user exits to be refreshed while DBRC is running.
- Setting traces is easier.
- Configuring DBRC is easier with BPE.

5.1.1 Setting up DBRC to run under BPE

To run DBRC under BPE requires task JCL and control statements.

DBRC task JCL using BPE

In the task JCL, you provide two parameters that identify members of PROCLIB:

- DBRCINIT, is the suffix of the DBRC initialization member DSPBIxxx.
- BPECFG is the name of the BPE control statements member.

An example of the BPE DBRC JCL is in member DSPBPROC of the SDFSISRC library.

The JCL we used in our system is discussed in B.1.3, "DBRC" on page 319.

DBRC initialization member DSPBIxxx

The parameters of this member are:

- IMSPLEX is needed for Automatic RECON Loss Notification (ARLN) or Parallel RECON Access (PRA).
- DBRCGRP allows you to separate different DBRC groups within an IMSplex. This parameter is needed for Parallel RECON Access.
- VSAMBUFF allows you to override the default DBRC buffers without modifying DSPBUFFS. Batch jobs cannot run under BPE, which means you must continue to modify DSPBUFFS if you want to override the RECON buffers in a batch job.

A better approach is to set IMSplex name and DBRC group by using the DBRC SCI registration exit (DSPSCIX0). This approach saves you from having to specify these parameters correctly on every job that requires DBRC.

Example 5-1 shows a typical DSPBIxxx member.

Example 5-1 Example of member DSPBIxxx

```
VSAMBUFF(INDEX=60,DATA=120)           /* VSAM buffers for RECON     */
```

A good description of the parameters is in *IMS Version 11 System Definition,* GC19-2444.

Also the IMS 11 Syntax Checker understands DSPBIxxx.

BPE control statements

There are no special requirements for DBRC. However, there are traces and exit points you might want to specify.

Example 5-2 shows how your DBRC BPE control statements might look. It includes statements for all the currently defined DBRC traces and statements for BPE and DBRC exit points. STATINTV controls how often the BPE statistics exits are called. In this example, we have specified a default value of 600 seconds.

Example 5-2 Sample BPE control statements for DBRC

```
...
STATINTV=600
...
TRCLEV=(ERR,HIGH,DBRC)              /* DBRC error trace            */
TRCLEV=(RQST,MEDIUM,DBRC,PAGES=10)  /* DBRC request trace          */
TRCLEV=(MODF,MEDIUM,DBRC)           /* DBRC module flow trace      */
TRCLEV=(GRPS,MEDIUM,DBRC)           /* DBRC group services trace   */
...
EXITMBR=(BPEEXIT0,BPE)              /* BPE  user exit definitions  */
EXITMBR=(DSPEXIT0,DBRC)             /* DBRC user exit definitions  */
```

In *IMS Version 11 System Definition,* GC19-2444, refer to the information about BPE configuration parameter member of the IMS PROCLIB data set.

5.1.2 Exploiting BPE based DBRC

After your DBRC address space is running under DBRC, you can begin exploiting the benefits of BPE. These are the ability to:

- Refresh the exits in flight.
- Have many exits of the same type.
- Collect DBRC statistics using a BPE statistics exit.

Existing DBRC user exits

If you use the existing RECON I/O exit (DSPCEXT0) or DBRC Command Authorization exit (DSPDCAX0), you may modify these to run with the BPE exit interface.

Note: Even if you run DBRC under BPE, you are not required to change your exits (although you may find that changes to the RECON record will force you to change DSPCEXT0 anyway).You can use the previous DBRC exit interface. In this case you do not name the exits in the BPE exit list.

These exits also run in batch jobs accessing DBRC. BPE is not available in these jobs and so they use the previous DBRC exit interface. The sample exits DSPCEXT1 and DSPDCAX0 in the SDFSSMPL library show how to cope with both interfaces in the same exit.

Alternatively, you can write different exits for the different interfaces. This might be the simpler approach if you want different logic in the batch than online.

When you run the DBRC address space under BPE, your exit can have any name you like; you are not restricted to the names DSPCEXT0 and DSPDCAX0.

Example 5-2 contains an EXITMBR statement that is referring to another PROCLIB member, which contains DBRC exit definitions. We have called this member DSPEXIT0. Example 5-3 shows this member. It contains both types of DBRC exit. For the RECON I/O exit we override the default abend limit of 1.

Example 5-3 Specifying DBRC user exits with BPE

```
EXITDEF(TYPE=RECONIO,EXITS=(ZDBRCIO0),ABLIM=8,COMP=DBRC)
EXITDEF(TYPE=SECURITY,EXITS=(ZDBRCSE0),COMP=DBRC)
```

In *IMS Version 11 System Definition,* GC19-2444, refer to the information about BPE exit list members of the IMS PROCLIB data set.

Note: The DBRC SCI registration exit DSPSCIX0 is not available with BPE. If you are using this exit, it does not change when you run DBRC under BPE.

BPE statistics exits

When running under BPE, DBRC can take advantage of BPE statistics exits. These exits can provide valuable information about how your DBRC address space is running. These exits also receive BPE information, but your code can easily recognize when it is being called with DBRC statistics. This might allow you to replace your existing DSPCEXT0 exit, if you were only using it to collect DBRC statistics.

Example 5-2 on page 149 contains an EXITMBR statement that refers to another PROCLIB member containing BPE exit definitions. In this example it is called BPEEXIT0, which is where you define BPE exits, including the statistics exit (or exits). Example 5-4 shows how a new statistics exit can be added to an existing BPE exit definition.

Example 5-4 Specifying BPE statistics exits

```
...
EXITDEF(TYPE=STATS,EXITS=(...,MYDSTAT0,...))
...
```

In *IMS Version 11 System Definition,* GC19-2444, refer to the information about BPE exit list members of the IMS PROCLIB data set.

If you are interested in pursuing this option, a simple BPE DBRC statistics exit is available to download. We describe this sample in C.3, “Sample BPE DBRC statistics exit” on page 332.

BPE INITTERM exits

BPE also has an exit point that is called at initialization and again at termination. You might find this useful in conjunction with your statistics exits.

Refreshing a user exit

Running the DBRC address space under BPE enables you to refresh your exits while DBRC is running. You use the z/OS MODIFY command. In Example 5-5, notice that you do not refresh individual exits, but all the exits of the type (or types) that you specify.

Example 5-5 Refreshing DBRC exits in flight

```
F IMB2DBRC,REFRESH USEREXIT NAME(RECONIO STATS)
```

You can also display the attributes of your user exits. A useful display field is TEXT, which reports the eye catcher if you have placed one at offset 4 in your module. Example 5-6 shows such a display command and Example 5-7 shows the response.

Example 5-6 Displaying user exit attributes

```
F IM2BDBRC,DIS USRX NAME(*) SHOW(SIZE TEXT)
```

Example 5-7 User exit attributes

```
BPE0030I EXITTYPE MODULE       SIZE TEXT                        DBRCIM2B
BPE0000I STATS    HHGSTAT0 00000008 ....                        DBRCIM2B
BPE0000I STATS    MYDSTAT0 00000300 .MYDSTAT0-08/05/09-20.00... DBRCIM2B
BPE0032I DIS USRX COMMAND COMPLETED DBRCIM2B
```

Note: The abilities to display information about and to refresh BPE exits are not new in IMS 11, but the ability to use them with DBRC is new in IMS 11.

For more information about these commands, refer to *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands,* SC19-2432.

Tracing with BPE

After you are running DBRC under BPE, you can change the tracing options with z/OS **MODIFY** commands.

Even if you have forgotten to define external trace data sets in your BPE configuration, you can add them dynamically and activate an external trace without closing DBRC, as follows:

1. Update the BPE configuration member to include an EXTTRACE statement.

   Example 5-8 shows an example of this. We have used the symbol &JOBNAME to distinguish different trace data sets belonging to different tasks.

   Example 5-8 Sample BPE EXTTRACE statement

   ```
   ...
   EXTTRACE (GDGDEF (DSN(IMS11M.BPETRACE.&JOBNAME.) BLKSIZE(32760)
                          SPACE(15) SPACEUNIT(TRK))
                     IOBUFS(10))
   ...
   ```

2. Change the trace options with a command like that in Example 5-9. The keyword OPTION(REREAD) tells BPE to reread the configuration member and it is this that allows you to add a trace data set dynamically.

   Example 5-9 Changing DBRC trace options

   ```
   F IM1BDBRC,UPD TRTAB NAME(MODF) OWNER(DBRC) EXTERNAL(YES) OPTION(REREAD)
   ```

You can do more things with the UPDATE TRACETABLE command. For details, refer to *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands,* SC19-2432.

After you have collected trace records, refer to information about BPE-based DBRC trace records, in *IMS Version 11 Diagnosis,* GC19-2436, to interpret them.

Again, this ability is not new with IMS 11, but it is new to DBRC.

5.1.3 New DBRC REQUEST user exit

The new DBRC REQUEST user exit has been added through APAR PK93338 (currently open). This exit is available to all IMS 11 BPE-based DBRC users.

DBRC REQUEST user exit provides an exit capability before and after a DBRC request is processed in BPE-based DBRC environment. It gives the option to call the user-written program before DBRC calls the request processing routine and after DBRC completes the request processing. The user also has the option to bypass DBRC request processing, however this option is not recommended because results are unpredictable if you choose to bypass DBRC request processing.

On the call to the DBRC Request User Exit, the pointer to DFSBRLSB and address of the IMS SCD will be passed. The use of the DBRC request user exit is optional.

The setup of this routine is the same as for BPE-based DBRC with the new EXITDEF type to DBRC User Exit List PROCLIB member. Refer to *IMS Version 11 Release Planning,* GC19-2442 for BPE-based DBRC setup.

5.2 Security override for non-production copies of RECON

Before IMS 11, a field in the RECON header controlled RECON security. This meant that if you had activated security, the same security applied to all copies of the RECON, even those copies that you had taken for testing or to diagnose a problem.

IMS 11 changes the security checking by adding a field to the RECON header. This field is named *RECON qualifier*. It is a string that must appear in the RECON COPY1 data set name for security to be active. If that string is not present in the RECON COPY1 data set name, then security is not active. This means that a copy of a production RECON is not protected if it has a sufficiently different name.

Using the RECON qualifier is good practice if you activate RECON security. If you do not specify a RECON qualifier, all copies of the RECON will be protected.

The two ways to use the RECON qualifier are:

- If the RECON qualifier ends with an asterisk *and* is enclosed in single quotation marks (‘*’), IMS checks that the RECON data set name *starts* with the RECON qualifier.
- Otherwise, IMS checks that the RECON data set name *contains* the RECON qualifier.

Example 5-10 shows how you might activate DBRC security. In this example, we have asked for SAF security using profiles that start with IM0B but that security only be active for RECON data sets that have names starting with:

`‘IMSPSA.IMOB.’`

So, for example, a copy named `‘IMSPSA.IMOB.RECON1.BACKUP’` would be protected, but a copy called `‘COPY.IMSPSA.IMOB.RECON1’` would not.

Example 5-10 Activating DBRC security, name starting

```
CHANGE.RECON CMDAUTH(SAF,IMOB,’IMSPSA.IMOB.*’)
```

If we had not specified an asterisk (*) in the RECON qualifier (see Example 5-11), then we would not have needed quotation marks and security would instead be active for RECON data sets whose name *contains*:

```
'IMSPSA.IMOB'
```

In this case, a copy called `'IMSPSA.IMOB.RECON1.BACKUP'` would be protected, and a copy called `'COPY.IMSPSA.IMOB.RECON1'` would also be protected.

Example 5-11 Activating DBRC security, name containing

```
CHANGE.RECON CMDAUTH(SAF,IMOB,IMSPSA.IMOB)
```

5.3 Removing old data from the RECONs

IMS 11 adds the following DBRC command, which you can use to remove old data from the RECON:

```
CLEANUP.RECON
```

Removing old data is necessary because PRILOG compression and the following command do not always remove old data:

```
DELETE.LOG INACTIVE
```

For example, you might have records for logs, which are still open because a batch job failed without closing them.

The CLEANUP.RECON command is also useful if you have a process that deletes data sets but leaves them registered in DBRC.

5.3.1 Considerations for using CLEANUP.RECON

You should activate DBRC command security for this CLEANUP.RECONcommand because it can do damage if used incorrectly.

Consider using CLEANUP.RECON first on a copy of the RECONs for two reasons:

- To protect yourself against unintended deletions
- To estimate the amount of time to process the command. (You would not want to run a large deletion from the production RECONs at a busy time.)

After you have deleted amounts of data, you might want to reorganize your RECONs to ensure that the freed space can actually be reused.

You must issue CLEANUP.RECON through a batch DBRC job or the DBRC API. It is not available from IMS through an **`/RMx`** command.

5.3.2 Using CLEANUP.RECON

Example 5-12 shows the syntax of the command. You specify a deletion time, either directly or by specifying a retention period from which DBRC calculates a deletion time.

Optionally you can specify:

- A range of databases to process
- That only database records be deleted.

  Log records are not deleted or compressed in this case.
- That DBRC should delete the last available image copy (use this only for databases that you know to be redundant).
- Whether DBRC should report, in detail, which records are deleted.

Example 5-12 cLEANUP.RECON syntax

```
CLEANUP.RECON TIME(...) | RETPRD(...) | LOGRET value from RECON header
              DBRANGE(firstdb,lastdb) | DBRANGE(firstdb) | DBRANGE(,lastdb)
              DBONLY
              LASTIC
              LISTDL | NOLISTDL | LISTDL value from RECON header
```

The CLEANUP.RECON command deletes:

- Allocation records older than the deletion time
- Image copy records older than the deletion time

  The command does not delete the last available image copy for a database unless you explicitly ask for this by using the LASTIC parameter.
- PRILOG records that started before the deletion time and are not required for recovery (no active subsystem and no LOGALL records)

  IMS compresses PRILOG records that cannot be deleted, but does not compress entries newer than the deletion time.
- Recovery records older than the deletion time
- Reorganization records older than the deletion time

The CLEANUP.RECON command deletes records that are not normally deleted; it does not consider GENMAX or RECOVPD.

Example 5-13 on page 155 shows part of the output from a CLEANUP.RECON command submitted on our system. These are examples of the different types of records being deleted.

Example 5-13   Part of a CLEANUP.RECON report

```
   CLEANUP.RECON RETPRD(007) LISTDL
...
DSP0123I  DBDNAME=DI21PART  DDNAME=DI21PARO
DSP1214I  RECON INFORMATION WAS DELETED FOR DBNAME=DI21PART  DDN=DI21PARO
DSP1214I    RECORD    TIME
DSP1214I    ------    ----
DSP1214I    IMAGE     2009.212 18:54:01.735491 -04:00
DSP1214I                 IC1 DSN=IMSPSA.IMOB.DI21PART.DI21PARO.IC103
DSP1214I    IMAGE     2009.212 18:57:15.135747 -04:00
DSP1214I                 IC1 DSN=IMSPSA.IMOB.DI21PART.DI21PARO.IC101
DSP1214I    ALLOC     2009.218 17:03:12.258072 -04:00
DSP0123I  NO PREDEFINED IMAGE COPY DATA SETS REMAINING
DSP0123I  DBDNAME=DI21PART  DDNAME=DI21PART
DSP1214I  RECON INFORMATION WAS DELETED FOR DBNAME=DI21PART  DDN=DI21PART
DSP1214I    RECORD    TIME
DSP1214I    ------    ----
DSP1214I    IMAGE     2009.212 18:54:02.422516 -04:00
DSP1214I                 IC1 DSN=IMSPSA.IMOB.DI21PART.DI21PART.IC102
DSP1214I    IMAGE     2009.212 18:57:15.787979 -04:00
DSP1214I                 IC1 DSN=IMSPSA.IMOB.DI21PART.DI21PART.IC103
DSP1214I    ALLOC     2009.218 17:03:10.699727 -04:00
...
DSP1216I  THE PRILOG FAMILY WITH TIME=2009.219 11:52:49.500455 -04:00 AND SSID=IM2B WAS
DELETED
DSP1047I  DELETED DSN=IMSPSA.SLDSP.IM2B.D09219.T1152495.V11,
DSP1047I  FILESEQ=0001,VOLSER=TOTIMY
DSP1047I  DELETED DSN=IMSPSA.SLDSP.IM2B.D09219.T1152495.V11,
DSP1047I  FILESEQ=0001,VOLSER=TOTIMY
...
DSP1150I  LOG RECORD(S) COULD NOT BE COMPRESSED,
DSP1150I  RECORD TIME = 2009.219 11:57:54.575837 -04:00
DSP1150I  EARLIEST ALLOC TIME  = 2009.219 11:57:55.580543 -04:00
DSP0133I  PRILOG IS ASSOCIATED WITH AN ACTIVE SUBSYSTEM
DSP0133I  RECORD TIME=2009.219 11:59:36.758334 -04:00
DSP0133I  SSIDNAME=IM1B
...
DSP0126I  NUMBER OF INACTIVE PRILOG RECORDS DELETED WAS 00006
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 04
DSP0220I  COMMAND COMPLETION TIME 2009.231 13:55:23.224224 -04:00
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 04
```

For more examples of the CLEANUP.RECON command, refer to *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands,* SC19-2432.

5.4 DBRC migration to and coexistence with IMS 11

Migration and coexistence information is provided in Chapter 7, “Installation and migration considerations” on page 227. This sections discusses several points about coexistence and migration.

5.4.1 Coexistence

Different coexistence considerations apply to different IMS versions.

Coexistence with IMS 9

An IMS 9 system can coexist with IMS 11, if:

- The RECON has been upgraded to IMS 11.
- APAR PK61582 (PTF UK42649) is applied.
- MINVERS in the RECON data set has not been set higher than 9.1.

  This settings means that no functions introduced by either IMS 10 or IMS 11 are available. Specifically not available are:

  – Parallel RECON access
  – Separation of DBRCs within an IMSplex by DBRC group

    IMS 9 processes automation RECON loss notifications sent by any member of the IMSplex, regardless of the group
  – Increased time-stamp precision
  – Database quiesce
- Programs using the DBRC API that were assembled with IMS 9 continue to run but cannot use enhancements or new fields that were introduced in IMS 10 or IMS 11.

  If you want to use these enhancements, you should make the necessary changes and reassemble with the IMS 11 macro library SDSFMAC.

Coexistence with IMS 10

An IMS 10 system can coexist with IMS 11, if:

- The RECON has been upgraded to IMS 11.
- APAR PK61583 (PTF UK42503) is applied.
- MINVERS in the RECON data set has not been set to 11.1.

  This means that database quiesce is not available.

5.4.2 Migration

As always, you must upgrade RECON to IMS 11 before starting an IMS 11 control region.

If you want to run DBRC under BPE, you should follow the steps described in 5.1.1, “Setting up DBRC to run under BPE” on page 148. You may find it convenient to keep the old non-BPE JCL in case you need to fall back.

As usual with new IMS versions, RECON records and listings have changed, so your own programs that are using them might have to change. Note the following information:

- The new RECON qualifier makes the RECON header longer.
- The change accumulation execution record has increased in size for future use.
- The database/area authorization record has increased in size for future use.
- New authorization flag values are associated with database quiesce.

  If you have a DBRC API program that uses these values, reassemble the program with the IMS 11 macros.

6

Connectivity and on-demand enhancements

IMS was and is a strong player as a data manager and a transaction manager in the service-oriented architecture (SOA). Many programs have been developed for accessing and processing data from traditional terminals or from batch. This rich set of assets can be opened up to the SOA world. The software component required for this support is IMS Connect, which is currently delivered with the IMS product distribution. IMS Connect has evolved a lot over time. Again in IMS 11, an additional gateway is opened, which allows for easy remote and local access to the DLI databases managed by the database manager component of IMS(DBCTL), without passing through an IMS transaction.

In this chapter, we introduce the role of IMS Connect, and the required configuration for supporting it. We also give an overview of other SOA-related usage of the IMSCON function and the related technology. Several SOA technologies (both existing and improvement in IMS 11) are packaged in a new product, the *IMS Enterprise Suite*. IMS Enterprise Suite V1.1 (program numbers 5665-T60 and 5665-T61) contains the following items:

- SOAP Gateway; see 6.2, “IBM IMS Enterprise Suite SOAP Gateway” on page 180
- Connect API for Java and C; see 6.1.5, “IMS Enterprise Suite V1.1 Connect API for Java and C” on page 178
- JMS API; see 6.4, “IBM IMS Enterprise Suite JMS API” on page 192
- DLIModel; see 6.5, “IMS as a database (DLI) server in SOA” on page 193
- SMP/E and Install Manager (Eclipse) support

This chapter contains the following topics:

- IMS Connect and new functions
- IBM IMS Enterprise Suite SOAP Gateway
- IBM IMS Enterprise Suite DLIModel utility
- IBM IMS Enterprise Suite JMS API
- IMS as a database (DLI) server in SOA
- IMS (DCCTL) as a service provider in SOA
- IMS (DCCTL) as a service consumer in SOA

6.1 IMS Connect and new functions

As shown in Figure 6-1, IMS uses IMSCON on the *call-in* side and *call-out* side. With this usage, the IMS component DCCTL(TM)[1] can function as both a *service provider* and a *service consumer*. The IMS component DBCTL(DB) can also be a *service provider* for DLI database access. The latter function, already existing in a limited way in IMS 10, has been enhanced in IMS 11 with the Open Data Base Manager (ODBM) component. This feature, which is easy to use and install, opens up the DLI databases for the service-oriented approach, as with other database systems.

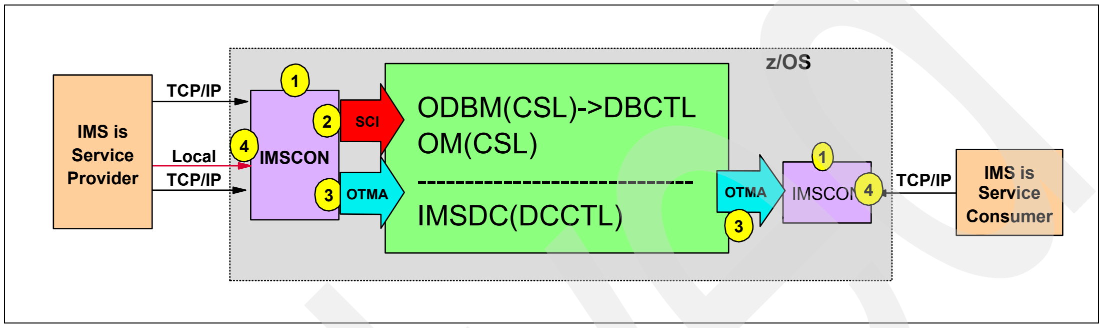


Figure 6-1 IMS as a service provider and consumer

Notes:

- The provider and consumer functions can be shared by the same IMS Connect.
- The SCI protocol is used for connecting to CSL address spaces (OM, ODBM). SCI can work in both PC and XCF mode
- OTMA is used for connecting to DCCTL. This also works across LPARs.
- From a TCP/IP point of view, IMS Connect is always the server (call-in and call-out).

As a service provider, the combination IMS Connect and IMS has three roles (all are as *call-ins*):

- Access to IMS transactions from J2EE and non-J2EE clients

  This is the first function to be made available in IMSCON. Access from a client goes over TCP/IP to IMSCON. Local access over program call is also possible. From IMSCON to IMS the access is over OTMA.

- Control function of IMS through DB2 Control Center installed on a workstation

  From IMSCON, the access to the Common Service Layer (CSL)/Operation Manager (OM) address space is over the SCI protocol. You can manage your IMS systems in a graphical interface from a workstation by using the IBM DB2 V9.1 for Linux®, UNIX®, and Windows® Control Center. Using the Control Center for IMS provides a single point of control that simplifies system management tasks in an IMSplex environment

- Access to DLI databases from J2EE and non-J2EE clients

  From IMSCON access to the CSL/Open Database Base Manager (ODBM) address space is over the SCI protocol. The ODBM address space, introduced in IMS 11 is a participant in the Common Service Layer communication.

[1] The DCCTL environment is an IMS Transaction Manager subsystem that has no database components. A DCCTL environment is similar to the DC component of a DB/DC environment.

As a service consumer, the combination IMS and IMS Connect allows for call-outs from programs running in Message regions (batch and non-batch). We distinguish the following types of call-outs:

- Synchronous call-outs

  Synchronous call-outs (also available through an IMS Version 10 SPE) for IMS. The Transaction Manager Resource Adapter (TMRA) or the SOAP gateway manage the correlation of a call-out request with the response. This approach enables IMS applications to invoke external applications and synchronously receive a response in the same IMS transaction instance. External applications and servers include Java EE applications, such as Enterprise JavaBeans (EJB) and Message-Driven Beans (MDB), deployed in a WebSphere Application Server, and Web services.

- Asynchronous call-outs

  In asynchronous call-out mode, the IMS application calls out to an external application and terminates. The output (if any) is sent back to another IMS application instance to continue processing. The advantage of using asynchronous call-out is that it does not hold up the region, but it does require IMS applications to be designed in a way that they are able to handle the output (response) from the external application in a different IMS application instance.

6.1.1 Enhancements in IMS 11 for IMS Connect

Enhancements exist for both for IMSDC and IMSDB, but the possibility to access DLI databases is probably the most rewarding.

IMS Connect enhancements for IMS DB

Enhancements include:

- IMS Connect is the TCP/IP path into IMS DB, IMS TM. IMS Connect clients can access IMS DB by using the open standard Distributed Relational Database Architecture (DRDA) specification, which supports distributed data management (DDM) architecture commands.
- Support for the IMS DB Universal drivers, the new CSL Open Database Manager (ODBM), and the DRDA communications protocol.

IMS Connect enhancements for IMS DC

Enhancements include:

- Expanded ACK timeout support for commit-then-send (CM0) output

  OTMA has expanded the ACK timeout support in IMS Version 11 to include commit-then-send (CM0) output. When a CM0 timeout occurs, the message is placed on a default timeout queue (named DFS$$TOQ). IMS Connect allows customers to specify what the CM0 timeout queue should be named to override the default timeout queue name.

- Improvements to connections include:
  - A keepalive function enables you to override the stack value of the TCP/IP keepalive parameter value on each port.
  - A Cancel Client ID function enables the client application to request a cancel-client-D when establishing a new connection with IMS Connect using the same client ID. The

previous session is discarded and the session continues without creating a duplicate client ID.

  – A TCP/IP auto reconnect feature, enables IMS Connect to continue to listen on the LISTEN SOCKET for the re-emergence of the network.

- The ability to specify a connection-level *super member*.

  IMS Connect client applications can share asynchronous output by using the OTMA super member function, which joins the tpipe queues of participating IMS Connect instances into a common shared super member group.

  By using the data store-level specifications for super member groups, application architects can create one or more subgroups of data stores that share asynchronous IMS output within a single instance of IMS Connect.

- An improvement to the reliability of diagnostic information, through exploitation of the BPE External Trace facility.
- Changes to the recorder trace process:
  – Extension of the Base Primitive Environment (BPE) External Trace facility. The extension, BPE Direct External Trace, enables BPE to write ad-hoc data of variable length directly to a copy buffer.
  – Exploitation of the BPE Direct External Trace facility. The recorder trace exploits the BPE Direct External Trace facility by rerouting recorder trace data to the BPE External Trace data set.
- Additional information in the HWSP1410W message

  To help diagnose problems that might occur when IMS Connect frees storage used for buffers, IMS 11 adds the address of the buffer that is associated with the error to the existing message HWSP1410W.

- Enforcement of the single port requirement for Secure Sockets Layer (SSL) sockets

  Prior to IMS 11, if users specified more than a single SSL port for an instance of IMS Connect, a failure could occur in the IBM Language Environment® for z/OS. In IMS 11, IMS checks for the correct specification of SSL ports. If multiple SSL ports are specified, or if the SSL port specification exceeds eight characters, IMS Connect issues a message and abends. If users need multiple SSL ports for their IMS Connect connections, they can transfer the management of the SSL ports from IMS Connect to the IBM z/OS Communications Server Application Transparent Transport Layer Security (AT-TLS).

  Note: Application Transparent TLS (AT-TLS) is a unique usage of TLS on the z/OS end of the session. Instead of having the application itself be TLS-capable and TLS-aware, the establishment of the TLS connection is pushed down the stack into the TCP layer.

  Many applications on z/OS can run without even being aware that the connection is using TLS. Remote clients cannot distinguish between *normal* TLS (where the application is doing the socket calls necessary for TLS) and AT-TLS (where the TCP layer handles the connection).

  Because TCP/IP is a layered protocol, the changes done at the TCP layer are hidden from the application layer. AT-TLS appears identical to normal TLS to any application connecting to the z/OS host.

- An option for leaving a connection open after returning a user-defined message from an IMS Connect exit routine

- The ability to modify input messages from TCP/IP before they are submitted to IMS Connect

  An exit routine, IMS Connect Port Message Edit exit routine, is introduced to provide you with an exit between TCP/IP and IMS Connect where you can modify the format of incoming and outgoing messages. The exit is specified on the IMS Connect port parameter so that you can use specialized exits on different ports as required by various IMS Connect clients. If a message coming from an IMS Connect client does not conform to the standard IMS Connect message format, you can modify the message format (using an exit routine) before the message comes into IMS Connect and modify it again after IMS Connect finishes processing it. In this way, you can make messages compatible with both the client and IMS Connect.

  The IMS Connect exit parameter list (HWSEXPRM) is changed for IMS 11. You must reassemble and rebind the IMS Connect exit routines that use HWSEXPRM to pick up the changes.

- Information listings are improved through optional summary versions of the VIEWHWS and QUERY MEMBER commands

  Summary versions of the VIEWHWS and QUERY MEMBER commands are provided to bypass listing each individual client for the ports. The data store name on the client output for the ports is added to indicate the IMS that the transaction was routed to.

- Elimination of the requirement to use different IMS Connect ports for instances of distributed WebSphere applications

  IMS Connect can generate a client ID for the IMS TM Resource Adapter, thus eliminating the requirement to use different IMS Connect ports for instances of distributed WebSphere Application Server. IMS Connect can use the generated client ID to ensure the uniqueness of each socket and allow all instances of WebSphere Application Server to specify the same IMS Connect TCP/IP port.

- A new IMS Request Message (IRM) flag

  A new IMS Request Message (IRM) flag is added so that the transaction expiration time can be passed in the OTMA message prefix.

- A warning message when the number of sockets is approaching the maximum specified

  IMS Connect currently supports a range of 50 - 65,535 sockets. The maximum number sockets that a particular instance of IMS Connect can support is specified in the MAXSOC parameter in the IMS Connect configuration member. When the number of sockets reaches the MAXSOC limit, IMS Connect refuses any new connections. The WARNSOC parameter can be used to specify a percentage of the MAXSOC limit. When the number of sockets reaches this percentage of the maximum, IMS Connect issues a warning message.

- Utilization of the new OTMA resource monitoring enhancement

  IMS Connect uses this new OTMA function by processing the protocol messages, updating its data store entry, and recording new data store events for warning and severity status. IMS Connect clients and user exits can access this information and redirect the transaction requests to a different IMS as necessary.

Commands for IMS Connect

Several new and changed IMS Connect commands are available, mostly to support Open Database. Using these commands you can:

- Query, stop, and start connections between IMS Connect and ODBM.
- Query, stop, and start ODBM alias names.
- Change whether IMS Connect automatically connects to ODBM.

Most commands have two formats: WTOR and z/OS MODIFY. Remember that you send BPE commands to IMS Connect using the z/OS MODIFY command.

More information about these IMS Connect commands is in *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands,* SC19-2432.

Displaying connections between IMS Connect and ODBM

To display connections between IMS Connect and ODBM, use one of the existing commands:

- `nnnn,VIEWHWS`
- `nnnn,VIEWDS data-store-name | ALL`
- `F xxxxxxxx,QUERY MEMBER TYPE(IMSCON)`

These commands have been enhanced to show ODBM information.

For example, if you reply to IMS Connect's WTOR **`nnnn,VIEWDS ALL`** command, you might see the following response:

```
DATASTORE=IM2B     STATUS=ACTIVE
...
ODBM=IM1BOD   STATUS=REGISTERED   ODBMRRS=Y
  ALIAS=IMOB     STATUS=ACTIVE
  ALIAS=IM1B     STATUS=ACTIVE
ODBM=IM2BOD   STATUS=REGISTERED   ODBMRRS=Y
  ALIAS=IMOB     STATUS=ACTIVE
  ALIAS=IM2B     STATUS=ACTIVE
```

Modifying connections between IMS Connect and ODBM

To change the state of an ODBM connection, you use one of these commands:

- `nnnn,STARTOD odbm-name | *`
- `nnnn,STOPOD odbm-name | *`
- `F xxxxxxxx,UPDATE ODBM NAME(odbm-name | *) START(COMM) | STOP(COMM)`

If you have not set up IMS Connect to connect automatically, you will have to issue commands every time you start ODBM.

For example you might enter the following commands at a z/OS console:

- `F IM2BCONN,UPDATE ODBM NAME(IM2BOD) STOP(COMM)`

  The response is:

  ```
  HWSN1960I ODBM=IM2BOD   TRANSMIT THREAD TERMINATED; M=NXMT
  HWSN1960I ODBM=IM2BOD   RECEIVE  THREAD TERMINATED; M=NREC
  HWSN1985I COMMUNICATION WITH ODBM=IM2BOD   CLOSED;        M=DSCM
  ```

- `F IM2BCONN,QUERY MEMBER TYPE(IMSCON) SHOW(SUMMARY)`

  The response is:

  ```
  HWS ID=IMOBHWS2 RACF=N  PSWDMC=N
  ...
    ODBM AUTO CONNECTION=Y
    ODBM TIMEOUT=18000
  ...
    ODBM=IM1BOD   STATUS=REGISTERED   ODBMRRS=Y
      ALIAS=IMOB     STATUS=ACTIVE
      ALIAS=IM1B     STATUS=ACTIVE
    ODBM=IM2BOD   STATUS=NOT ACTIVE   ODBMRRS=
  ...
  PORT=7000     STATUS=ACTIVE     KEEPAV=0 NUMSOC=1 EDIT=         TIMEOUT=0
  ```

```
  NO ACTIVE CLIENTS
PORT=7001     STATUS=ACTIVE     KEEPAV=0 NUMSOC=1 EDIT=        TIMEOUT=0
  NO ACTIVE CLIENTS
PORT=6000D    STATUS=ACTIVE     KEEPAV=0 NUMSOC=1 EDIT=        TIMEOUT=18000
  NO ACTIVE CLIENTS
PORT=6001D    STATUS=ACTIVE     KEEPAV=0 NUMSOC=1 EDIT=        TIMEOUT=18000
  NO ACTIVE CLIENTS
```

Notice that the ODBM ports end in the letter D to distinguish them from the transaction ports.

Displaying ODBM aliases

Several commands have been introduced to display the alias names that IMS Connect uses to connect to ODBM. You define the aliases in PROCLIB member CSLDCxxx.

The commands are:

- `nnnn,VIEWIA alias-name | *`
- `nnnn,VIEWDS data-store-name | ALL`
- `nnnn,VIEWHWS`
- `F xxxxxxxx,QUERY ALIAS NAME(alias-name | *)`

For example you might enter the following command at a z/OS console:

```
F IM2BCONN,QUERY ALIAS NAME(*)
```

You receive the following response:

```
ALIAS=IM0B  ODBM=IM1BOD   STATUS=ACTIVE
ALIAS=IM1B  ODBM=IM1BOD   STATUS=ACTIVE
ALIAS=IM0B  ODBM=IM2BOD   STATUS=ACTIVE
ALIAS=IM2B  ODBM=IM2BOD   STATUS=ACTIVE
```

Stopping and starting ODBM aliases

The commands to change the state of an ODBM alias are:

- `nnnn,STARTIA alias-name | *`
- `nnnn,STOPIA alias-name | *`
- `F xxxxxxxx,UPDATE ALIAS NAME(alias-name | *) START(ROUTE) | STOP(ROUTE)`

For example you might issue the following commands:

1. `nnnn,STOPIA IM0B`
2. `nnnn,VIEWIA *`

The response is:

```
ALIAS=IM0B  ODBM=IM1BOD   STATUS=NOT ACTIVE(IMSCON)
ALIAS=IM1B  ODBM=IM1BOD   STATUS=ACTIVE
ALIAS=IM0B  ODBM=IM2BOD   STATUS=NOT ACTIVE(IMSCON)
ALIAS=IM2B  ODBM=IM2BOD   STATUS=ACTIVE
```

To stop this alias for only one of the ODBMs, we can use the **`nnnn,STOPIA IM0B IM2BOD`** command.

Changing automatic connection to ODBM

Use the commands to change whether or not IMS Connect connects automatically to ODBM:

- `nnnn,SETOAUTO YES | NO`
- `F xxxxxxxx,UPDATE MEMBER TYPE(IMSCON) SET(OAUTO(ON | OFF))`

This is a global setting and therefore applies to all ODBMs.

6.1.2 Configuring IMS Connect

IMS Connect supports communication from one or more clients (TCP/IP and one local) to IMS systems. You can configure multiple IMS systems on multiple z/OS images within a single sysplex and distribute the client request to the IMS systems (data stores).

To make IMS Connect ready for usage, perform the following tasks:

- Authorize the Application Program Family (APF).

  IMS.SDFSRESL, the resident library in which the IMS Connect modules reside, must be authorized to the APF and used for IMSplex support.

- Update the z/OS Program Properties Table (PPT).

  Updating the PPT allows IMS Connect to run in authorized supervisor state and in key 7.

- Enable Internet Protocol Version 6 (IPV6) for IMS Connect:

  a. Customize the BPXPRMxx member, as shown in Example 6-1.

  Example 6-1 BPXPRMxx member contents

```
FILESYSTYPE Type(INET) Entrypoint(EZBPFINI)
NETWORK DOMAINNAME(AF_INET)
DOMAINNUMBER(2)
  MAXSOCKETS(2000)
  TYPE(INET)

NETWORK DOMAINNAME(AF_INET6)
DOMAINNUMBER(19)
  MAXSOCKETS(3000)
  TYPE(INET)
```

  b. For each of the READ subroutines that you use, determine whether the EXPREA_IPV6 bit is turned on in the EXPREA_FLAG2 field of the READ subroutine. If it is turned on, IPV6 is enabled. The READ subroutine is invoked in the user exits:

  - HWSJAVA0
  - HWSSMPL0
  - HWSSMPL1

  c. Map EXPREA_SOCKET6 to the AF_INET6 socket address structure.

- Create an IMS Connect configuration member to hold the configuration statements that IMS Connect uses during initialization.

  These statements are HWS, DATASTORE, TCPIP, IMSPLEX, ODACCESS, RUNOPTS, and ADAPTER.

  Customize the IMS Connect configuration member using the IPV6 parameter.

6.1.3 Configuration member (HWSCFGxx)

IMS Connect requires a configuration member to hold the configuration statements that IMS Connect uses during initialization. The IMS Connect configuration member defines how IMS Connect communicates with TCP/IP, OTMA, the CSL, Open Database Manager (ODBM), and security software.

In the definition statements, we highlight the following changes in IMS 11, mainly for the support of ODBM:

- HWS(): Defines characteristics specific to this instance of IMS Connect. Specify only one IMS Connect with subparameters. This statements has new subparameters in IMS 11.
  - ID: The IMS Connect name
  - PSWDMC= N/Y/R. Whether or not IMS Connect supports mixed-case passwords.
  - CM0ATOQ: Character name (1 - 8 characters) of the OTMA CM0 ACK timeout queues. The value specified here overrides the OTMA default value of DFS$$TOQ and is used for all data stores except those that have their own CM0ATOQ value (that is, if you specify a value for CM0ATOQ on the DATASTORE statement, it overrides a value for CM0ATOQ specified on the HWS statement).

    Note: You can specify a timeout interval value by using the ACKTO parameter in the DATASTORE configuration statement. The timeout value specified on the ACKTO parameter applies to acknowledgments sent to OTMA for both CM0 output and send-then-commit (CM1) output, and determines how long OTMA waits for an acknowledgment from IMS Connect.

    This timeout action frees the hung situation, releasing the original output queue to allow continued delivery.

    You can also specify a timeout tpipe queue to hold commit-then-send (CM0) output after the timeout interval has expired.

    For CM0 output, when the timeout interval expires, OTMA removes the output from the tpipe queue and reroutes the output to either a specified reroute tpipe queue, a specified timeout tpipe queue, or the default OTMA timeout tpipe queue DFS$$TOQ. The name of a timeout tpipe queue for CM0 output can be specified on the CM0ATOQ parameter in both the HWS and DATASTORE configuration statements. Commit mode 0 (CM0) is also called commit-then-send. CM0 is supported on both persistent and transaction sockets and supports only synch level CONFIRM.

    CM1 output is not rerouted to a timeout queue, because OTMA discards CM1 output if the timeout interval expires.

  - RRS: Specifies whether RRS communication is to be enabled or disabled.
  - RACF: At IMS Connect startup time, this determines whether the password and user ID (provided by either the client application or a user exit routine) are passed to RACF for authentication.

    Note: This parameter does not apply to the security support provided by IMS Connect for the CSL Open Database Manager

  - SMEMBER: This is the name (1 - 4 characters) of the OTMA super member to which this instance of IMS Connect belongs. The super member can also be specified on the DATASTORE statement. If a value is specified on both the HWS statement and the DATASTORE statement, the value on the DATASTORE statement takes precedence.

Note: Hold-queue-capable OTMA clients, such as IMS Connect, can share asynchronous commit-then-send (CM0) output messages by enabling the OTMA super member *function*. The OTMA super member function is specifically designed to support *multiple instances* of IMS Connect in a z/OS *Sysplex* Distributor environment.

The OTMA super member function allows for sharing of "asynchronous commit-then-send output (CM0)" by Hold-queue-capable OTMA clients, such as IMS Connect. The OTMA super member function manages all of the asynchronous CM0 output of all of its participating OTMA clients by using a common output queue. Any participating hold-queue-capable client can then retrieve the CM0 messages on the super member output queue by issuing its own RESUME TPIPE call, regardless of which client the CM0 output was originally destined for.

The OTMA super member support for *Shared Queues* provides the capability for an IMS Connect client to connect to any front-end IMS and retrieve an ALTPCB message that is created or will be created in any of the back-end systems in the *shared queues* group. The ability, using super member support to create a listening remote client that can wait for and retrieve these types of messages as they are queued, was previously documented as a restriction.

The messages on the super member queue are retrieved on a first-in-first-out basis, without regard to which instance of the *OTMA client* originated the output or which instance of the *OTMA client* issued the *RESUME TPIPE call*. The RESUME TPIPE call from a participating *OTMA client* does not need to specify anything about the super member. The super member function recognizes the RESUME TPIPE call as coming from a participating *OTMA client* and returns the output on the common queue.

Super Member name can now be specified on the data store level.

- XIBAREA: Number of full words allocated for the XIB user area.

- TCPIP()
  - ECB: Whether TCP/IP exit or ECB (Event Control Block) processing is to be used.
  - EXIT: Specifies the name (1 - 8 alphanumeric characters) of one or more IMS Connect user message exit routines that receive control when messages are received from and sent to TCP/IP clients (for example, HWSSOAP1 for "IBM IMS Enterprise Suite SOAP Gateway).
  - HOSTNAME: Specifies the TCP/IP JOBNAME
  - IPV6=Y/N. At IMS Connect startup time, this determines whether Internet Protocol Version 6 (IPV6) is enabled. Set this parameter to yes (Y) or no (N). If you leave this field blank, the default value N is used.

Note: If you use IPV6, the BPXPRMxx UNIX System Services member has to be adapted also and IMS Connect exits (HWSJAVA0, HWSSMPL0, HWSSMPL1) have to be verified.

  - MAXSIZE: Specifies the maximum message size that is allowed in the 4-byte length field that precedes the IMS request message (IRM). Use the MAXSIZE= keyword to override the internal default of 10000000 bytes.

– MAXSOC: This is a decimal value in the range of 50 - 65535 that sets the maximum total number of sockets that this instance of IMS Connect can open.

Note: The maximum number of physical connections that can be made is the value of MAXSOC minus the number of ports, because IMS Connect uses one socket on each port for listening. For example, if you specify MAXSOC=80 and have five ports, 75 physical connections can be made. The default value is 50.

When IMS Connect starts, each regular port has a socket count of 1, which reflects the port listen socket. The SSL port has a socket count of 2, which reflects one port listen socket and one SSL-related file descriptor. When the SSL port receives a connection for the first time, one additional SSL-related file descriptor is created and is counted towards the socket count.

When the number of sockets reaches the MAXSOC limit, IMS Connect refuses any new connections.

The MAXSOC parameter is related to the z/OS UNIX System Services parameter MAXFILEPROC. The values of MAXSOC and MAXFILEPROC must be compatible. If the values of each parameter are not compatible, IMS Connect cannot open any ports. The values are compatible when the value for MAXFILEPROC is equal to or greater than the value of MAXSOC.

You can ensure compatibility between MAXSOC and MAXFILEPROC by granting IMS Connect UNIX System Services superuser privileges, which allows IMS Connect to change the value of the MAXFILEPROC parameter automatically. You can grant UNIX System Services superuser privileges to IMS Connect by using the RACF command ALTERUSER to assign an OMVS segment with a UID of 0 to the user ID of the IMS Connect started task. Alternatively, your UNIX System Services administrator can adjust the value of MAXFILEPROC directly in the BPXPRMxx member of the z/OS SYS1.PARMLIB data set

– NODELAY: Whether (Y) or not (N) the TCP/IP protocol option TCP_NODELAY is to be used. The default is no.

– PORT: Specifies exit routines, enabling you to modify messages that do not conform to IMS Connect's standard message formats. You can identify specialized exits on different ports as required by IMS Connect clients, as follows:

- ID: Decimal field (1 - 8 characters) that defines the TCP/IP port
- KEEPAV: Number of seconds for the TCP/IP keepalive interval for sockets on this port

Note: The keepalive function, provided by the TCP/IP protocol, can detect certain socket error conditions by sending a keepalive packet on sockets that have been inactive for a specified interval. For example, the keepalive function can detect sockets that are no longer valid because the client has abruptly disconnected without informing IMS Connect.

By default, IMS Connect accepts the specification set in the z/OS layer for TCP/IP sockets; however, if a keepalive interval set by z/OS is large, as might be the case for installations trying to reduce network traffic, the large interval can delay the detection of an invalid socket by IMS Connect.

To help detect error conditions earlier on IMS Connect sockets, you can specify a smaller keepalive interval for IMS Connect ports. Each port defined to IMS Connect can specify a different keepalive interval. The keepalive interval specified for each port applies to all sockets that use that port.

- EDIT: Name (1 - 8 characters) of the Port Input/Output Edit Exit routine. The exit routine must be accessible to IMS Connect by either JOBLIB, STEPLIB, or LinkList. This field is optional. The default is no exit.

– PORTAFF: Specifies that commit-then-send (CM0) output messages that IMS sends to this IMS Connect have affinity to the port on which IMS Connect received the original input message.

– PORTID: A decimal field that defines TCP/IP ports (1 - 8 characters), or a 5-character field with the value of LOCAL to define the local option connection. You can define up to 50 ports.

– RACFID: Specifies default RACF ID for exits to pass to OTMA for security checking if the RACF ID has not explicitly been set in the incoming message or by the user exit.

– SSLENVAR: The TCP/IP variable SSLENVAR points to an SSL interface configuration file that contains variable assignment statements that provide the information necessary for SSL initialization. Both a HFS key database or RACF key ring are supported. If a RACF key ring is specified, it must be an existing key ring and the current user ID must have READ access to the IRR.DIGTCERT.LISTRING and the IRR.DIGCERT.LIST resources in the FACILITY class.

  When you configure the IMS Connect SSL interface, also consider the ability of the IMS Connect client to support SSL connections.

Note: SSL connections use public and private key mechanisms for authenticating each side of the SSL session (the server side and the client side) and agree on bulk encryption keys to be used for the SSL session. Authenticating both the client and server during SSL handshaking is not always necessary. In many cases, you can choose to generate a public and private key pair only on the server side to facilitate server authentication during SSL handshaking. The fact that the client sends a user ID and password over the secured connection after the SSL handshaking completes can be considered authentication of the client.

To use public and private key mechanisms (PKIs), public and private key pairs must be generated. In addition, X.509 certificates (which contain public keys) might either need to be created, or certificates must be requested, received, and managed. SSL for z/OS supports the following two methods for managing PKI private keys and certificates:

- A z/OS shell-based program named gskkyman, which creates, fills in, and manages a z/OS HFS file that contains PKI private keys, certificate requests, and certificates. This z/OS HFS file is referred to as a key database and, by convention, has a `.kdb` file extension.
- The z/OS Security Server (RACF) RACDCERT command. The RACDCERT command installs and maintains PKI private keys and certificates in RACF.

  RACF supports the management of multiple PKI private keys and certificates as a group. These groups are called key rings. RACF key rings are the preferred method for managing PKI private keys and certificates for SSL.

– SSLPORT: A decimal field (1 - 8 numeric characters) to define a Secure Sockets Layer (SSL) port. For SSL port communication, specify the port number that will bind to the socket. This port must not conflict with any other port that is selected in the TCP/IP domain or the ports that are selected under the PORT or PORTID parameter as basic TCP/IP ports.

Note: You can define only one SSL port. If you define more than one SSL port and you attempt to use another port, unpredictable results can occur. If you want to define and use more than one SSL port for IMS Connect, you can define additional SSL ports using the z/OS Communications Server. IMS considers these ports as additional PORTIDs on the IMS Connect TPCIP statement.

– TIMEOUT: Used by IMS Connect to determine the amount of time to wait for a response from IMS that is being sent to the client. This timeout value is used to prevent the client from appearing to be in a hang condition, which occurs when the IMS host application is not responding, because of one of the following reasons:

  - The IMS program for this transaction code is stopped.
  - The dependent region that would run the transaction is not active.
  - The IMS host application is looping.

– WARNINC: Specifies a warning level incremental percentage. After the WARNSOC value is reached, every time the number of sockets increases by the percentage value specified on WARNINC, IMS Connect reissues warning message HWSS0772W.

– WARNSOC: Specifies a warning level when the number of sockets increases to a certain percentage of the MAXSOC limit. When the number of sockets that are currently supported reaches this warning level, IMS Connect issues a warning message, HWSS0772W.

- DATASTORE()

  To access IMS OTMA, specify each DATASTORE with which the IMS Connect communicates through IMS OTMA:

  – APPL: This is a 1 - 8 alphanumeric character field set to the TCP/IP APPL name. If you are using PassTicket and user message exits, you must specify the APPL on the DATASTORE statement.
  – ACKTO: The timeout value specified that applies to acknowledgments sent to OTMA for both CM0 output and send-then-commit (CM1) output; determines how long OTMA waits for an acknowledgment from IMS Connect.
  – CM0ATOQ: See explanation in HWS(), which begins on page 165.
  – DRU: This is a 1 - 8 alphanumeric character field to specify your own OTMA destination resolution user exit name.
  – GROUP: The XCF group name for the IMS OTMA.
  – ID: The data store name. This ID must match the data store ID that is supplied by the client. For IMS TM Resource Adapter clients, this ID must match the name that is specified in the IMS Interaction Spec for IMS TM Resource Adapter. For non-IMS TM Resource Adapter clients, the ID must match the data store ID that is placed in the IMS request message (IRM) that is sent to IMS Connect.

    Note: If the IMSplex statement or ODACCESS statement is also specified, the data store name cannot be the same as the TMEMBER name on either statement.

  – MAXI: This is the OTMA input message flood control value.
  – MEMBER: The XCF member name that identifies IMS Connect in the XCF group specified by the GROUP parameter
  – SMEMBER: See explanation in HWS(), which begins on page 165.

    Note: The DATASTORE statement was enhanced to support the SMEMBER parameter for super member support at a data store level. The SMEMBER parameter on individual DATASTORE statements overrides the default HWS value. By providing the super member support at a data store level, IMS Connect can now support multiple super member names and have more granularity for the usage of the *super member* concept.

    Figure 6-2 on page 171 shows a configuration with two IMS Connects, both addressing the same IMS. Without the *super member* concept, USER1 connecting through IMSCON1, would be different from USER1 connecting through IMSCON2 and use a different output tpipe. Retrieval of CM0 messages that are accumulated through IMSCON1 by USER1, cannot be retrieved by USER1 through IMSCON2. This challenge is solved by the concept of the *super member*, where a common tpipe is used.

    Note: Prior to IMS 11, a super member could only be specified on the IMSCON level (HWS()); with IMS 11, it can be specified on the DATASTORE() statement.

    The usage of the super member is vital when the Sysplex Distributor is used in front of the two IMSCONs, in which case sysplex distributor with work load balancing decides which IMSCON will take the TCP/IP in request.

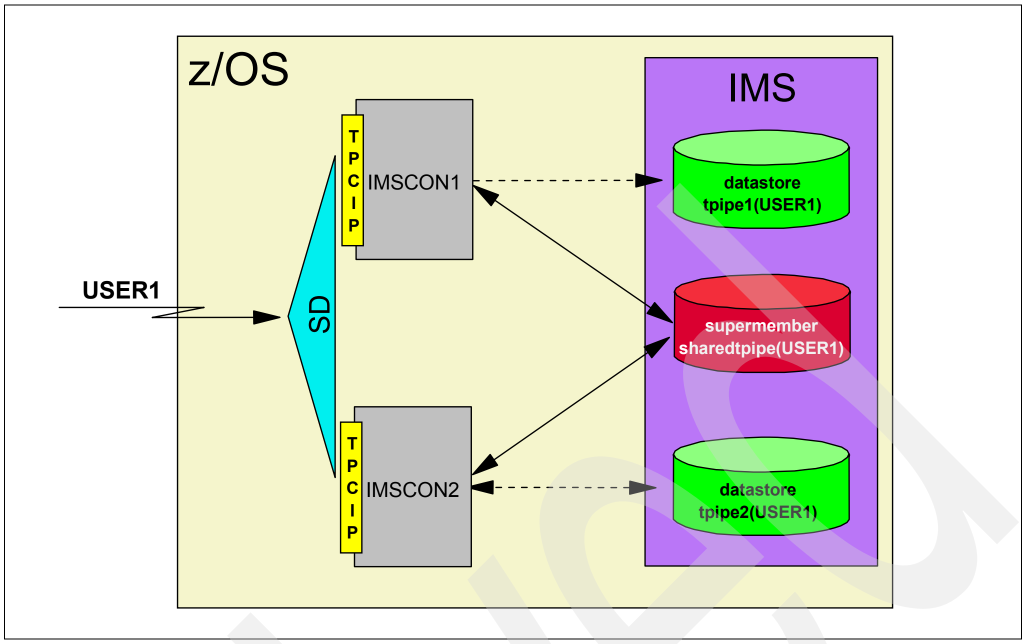


Figure 6-2 Super member usage with two IMSCONs to one IMS

- RRNAME: The name of an alternate destination specified in a client reroute request. If this string is not provided, IMS Connect uses HWS$DEF as the default name.
- OAAV: A decimal integer that defines the OTMA ACEE aging value, in seconds, for the data store that is specified by the ID keyword. When the OTMA ACEE aging value is reached, OTMA refreshes the RACF Accessor Environment Element (ACEE) before it processes the next input message received from IMS Connect.
- TMEMBER: The XCF member name for IMS that IMS Connect uses in order to communicate with an IMS in its XCF group. This target member name must match the member name IMS uses when it joins the XCF group.

- IMSPLEX()

  To access the IMS Operations Manager (OM), IMS Open Database Manager, or both. Specify one value for each IMSPLEX that IMS Connect communicates with through the IMS Structure Call Interface (SCI):

  - MEMBER: Specifies the name that is passed to the SCI as the name of the IMS Connect that is communicating with OM, ODBM, or both through the SCI.
  - TMEMBER: Specifies the name of the SCI to which IMS Connect communicates. The TMEMBER name:
    - Consists of alphanumeric character data
    - Begins with an alphabetic character
    - Has a length of 1 - 5 bytes
    - Must be the name specified in the SCI initialization PROCLIB member, `IMSPLEX(NAME=name)`
    - Cannot be the same name as the ID name on the DATASTORE statement.

- ADAPTER=()

  Defines characteristics of adapters that are used to convert XML input to COBOL:

  - XML=Y/N: Specifies whether XML adapter support should be enabled (Y) or not (N).

Note: The XML adapter is used to convert the user data in the response message into XML. IMS Connect then sends the output message to the IMS SOAP Gateway.

- RUNOPTS=()

  Defines the Language Environment runtime options to be used to override the IMS Connect default runtime options in support of SSL.

- ODACCESS=()

  Defines characteristics of the communication between ODBM, DRDA clients (such as the Open Database APIs), and IMS Connect. This statement in new in IMS 11.

Important: Specify only one ODACCESS statement.

IMSCON is able to address several ODBM address spaces, belonging to many unrelated IMSs (IMSPLEX or Standalone IMS). The communication always happens over an SCI protocol, and is required that all ODBM CSL address spaces belong to the same CSL group. The name of this group is referenced in the initialization member of the ODBM CSL address by the keyword IMSPLEX. In Example 6-2 it is PLEXB.

Example 6-2 Initialization member for SCI

```
*--------------------------------------------------------------------*
* SCI INITIALIZATION PROCLIB MEMBER FOR IMB1ODBM                     *
*--------------------------------------------------------------------*
ARMRST=Y,               /* SHOULD ARM RESTART RM ON FAILURE       */
IMSPLEX(NAME=PLEXB)     /* NAME OF THE PLEX CSLPLEXB              */
SCINAME=IM1B            /* SCI NAME (SCIID = IM1BSC)              */
*--------------------------------------------------------------------*
* END OF MEMBER                                                      *
*--------------------------------------------------------------------*
```

Specify one:

– DRDAPORT: Defines the port numbers, the TCP/IP keepalive value, and the timeout values for the ports that IMS Connect uses to provide access to IMS for client applications of the Open Database APIs and user-written DRDA client applications. You can define up to 50 ports. Definitions are:

  - ID: Specifies the port number of the port being defined by the DRDAPORT parameter. Port numbers are specified by a 1 - 5 character numerical value. Valid port numbers are 1 - 65 535. Port numbers must be unique among all ports used by IMS Connect within the TCP/IP domain, including the basic TCP/IP ports specified on the PORT or PORTID parameter and any Secured Sockets Layer (SSL) port specified on the SSLPORT parameter.
  - KEEPAV: Sets the interval (in a 1 - 8 character decimal field) after which the keepalive mechanism of the z/OS TCP/IP layer sends a packet on idle connections on this port to maintain the connections. TCP/IP accepts a range of 1 - 2 147460 seconds. Specify 0 (default) to use the keepalive value defined in the z/OS TCP/IP stack.
  - PORTTMOT: Defines the amount of time that IMS Connect waits for the next input message from a client application that is connected on a DRDA port before IMS Connect disconnects the client. The timeout interval is specified as a decimal integer in hundredths of a second.

Valid values for PORTTMOT are 0 - 2147483647 (X'7FFFFFFF'). The default is 18000 (3 minutes). A value of 0 disables the timeout function. Specifying a timeout value can avoid hang conditions when a client stops sending messages as expected.

– IMSPLEX: An optional parameter that specifies the IMSplex in which IMS Connect communicates with ODBM through the IMS Structured Call Interface (SCI). If not specified, messages from the IMS Universal drivers and DRDA clients are routed to ODBM in the IMSplex specified on the IMSPLEX configuration statement.

  The IMSPLEX statement keyword parameters include:

  - MEMBER: Specifies a name (1 - 16 characters) that identifies IMS Connect to SCI, which uses this name when managing communications between IMS Connect and ODBM.
  - TMEMBER: The name of the SCI that manages communications between IMS Connect and ODBM. Specify the same SCI name that is defined in the NAME parameter of the IMSPLEX keyword of the SCI initialization member of the IMS PROCLIB data set (CSLSIxxx). See Example 6-2 on page 172 and Example 6-3 on page 174.

    The name specified in TMEMBER must be different from the name specified on the ID parameter of the DATASTORE statement.

– ODBMAUTOCONN: Specifies whether IMS Connect automatically connects to new and existing instances of ODBM within an IMSplex.

– ODBMTMOT: Defines the amount of time that IMS Connect waits for *both*:

  - A response message on connections with ODBM
  - An initial input message after a socket connection is established on connections with a client application

Note: The timeout interval is specified as a decimal integer in hundredths of a second. Valid values for ODBMTMOT are a range of 0 - 2147483647 (X'7FFFFFFF'). The default value is 18000 (3 minutes). A value of 0 disables the timeout function

For connections with ODBM, specifying a timeout value can avoid hang conditions when an ODBM instance stops responding.

For connections to a client application, specifying a timeout value terminates a socket connection if a client does not send data after obtaining the socket connection before the ODBMTMOT value expires.

Configuration member example

The IMS system we are using for the testing is described in Appendix B, “Configuration and workload” on page 313. Make the following changes to your IMS Connect configuration member to be able to access ODBM:

1. If you have not already defined your IMS Connect system to your IMSplex (for example you might not be using the IMS component of the DB2 Control Center), you must add an IMSPLEX statement.
2. You must also add an ODACCESS statement.

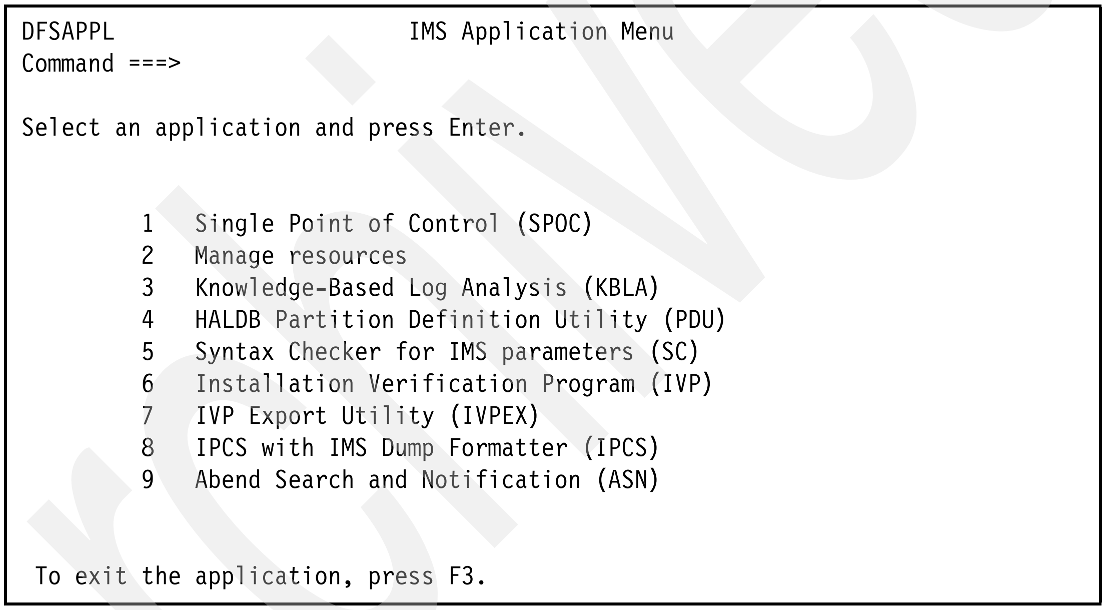
In Example 6-3, our IMSplex is called `PLEXB` and this IMS Connect is member `IM1BHWS1` within the IMSplex. We have defined two ODBM ports and requested that IMS Connect automatically connect to ODBM.

Notice that we do not have to tell IMS Connect what the ODBM address space (or spaces) are called. When an ODBM joins the IMSplex, IMS Connect will recognize it.

Example 6-3 ODACCESS and IMSPLEX statements for accessing ODBM from IMS Connect

```
...
IMSPLEX=(MEMBER=IM1BHWS1,TMEMBER=PLEXB)
ODACCESS=(DRDAPORT=(ID=6000,KEEPAV=0,PORTTMOT=18000),
          DRDAPORT=(ID=6001,KEEPAV=0,PORTTMOT=18000),
          ODBMAUTOCONN=Y)
...
```

Before using the member with IMS Connect, lets execute a syntax checking. Start with the IMS Application Menu, as shown in Figure 6-3.



```
DFSAPPL                      IMS Application Menu
Command ===>

Select an application and press Enter.


         1   Single Point of Control (SPOC)
         2   Manage resources
         3   Knowledge-Based Log Analysis (KBLA)
         4   HALDB Partition Definition Utility (PDU)
         5   Syntax Checker for IMS parameters (SC)
         6   Installation Verification Program (IVP)
         7   IVP Export Utility (IVPEX)
         8   IPCS with IMS Dump Formatter (IPCS)
         9   Abend Search and Notification (ASN)


 To exit the application, press F3.
```

Figure 6-3 IMS Application Menu

Select option **5**, **Syntax Checker for IMS parameters** and press Enter. The IMS Parameter Syntax Checker panel opens, as shown in see Figure 6-4 on page 175.

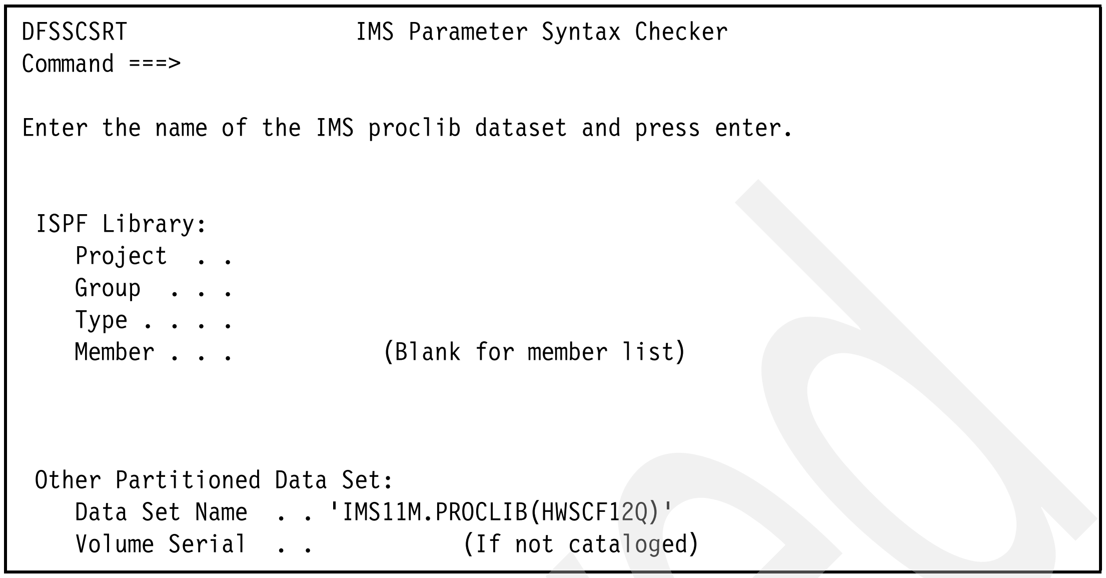

```
DFSSCSRT                 IMS Parameter Syntax Checker
Command ===>

Enter the name of the IMS proclib dataset and press enter.


 ISPF Library:
    Project  . .
    Group  . . .
    Type . . . .
    Member . . .           (Blank for member list)



 Other Partitioned Data Set:
    Data Set Name  . . 'IMS11M.PROCLIB(HWSCF12Q)'
    Volume Serial  . .           (If not cataloged)
```

Figure 6-4 IMS Parameter Syntax Checker

Enter the name of the HWSconfig member you want to verify, and press Enter.

The next panel opens, as shown in Figure 6-5.

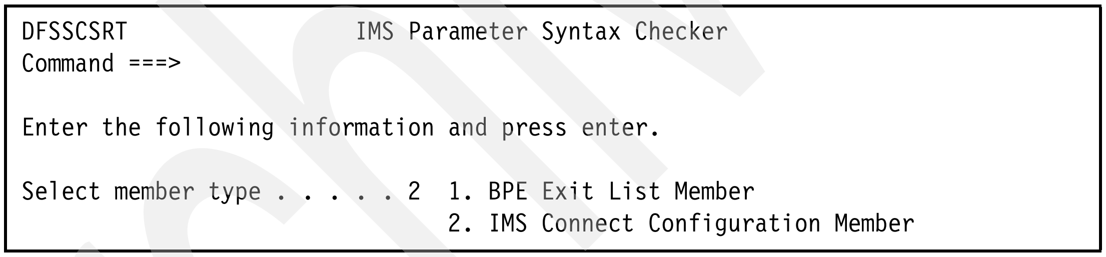

```
DFSSCSRT                 IMS Parameter Syntax Checker
Command ===>

Enter the following information and press enter.

Select member type . . . . . 2  1. BPE Exit List Member
                                2. IMS Connect Configuration Member
```

Figure 6-5 Select member type

Select the member type, which in our case is **2** (**IMS Connect Configuration Member**), and press Enter.

The next panel opens, as shown in Figure 6-6.

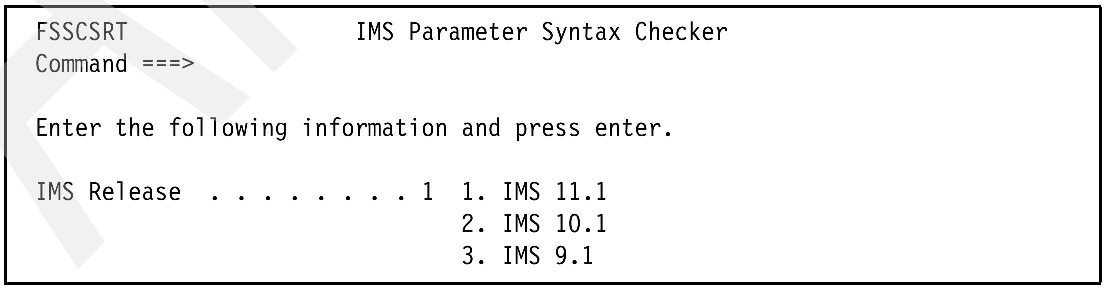
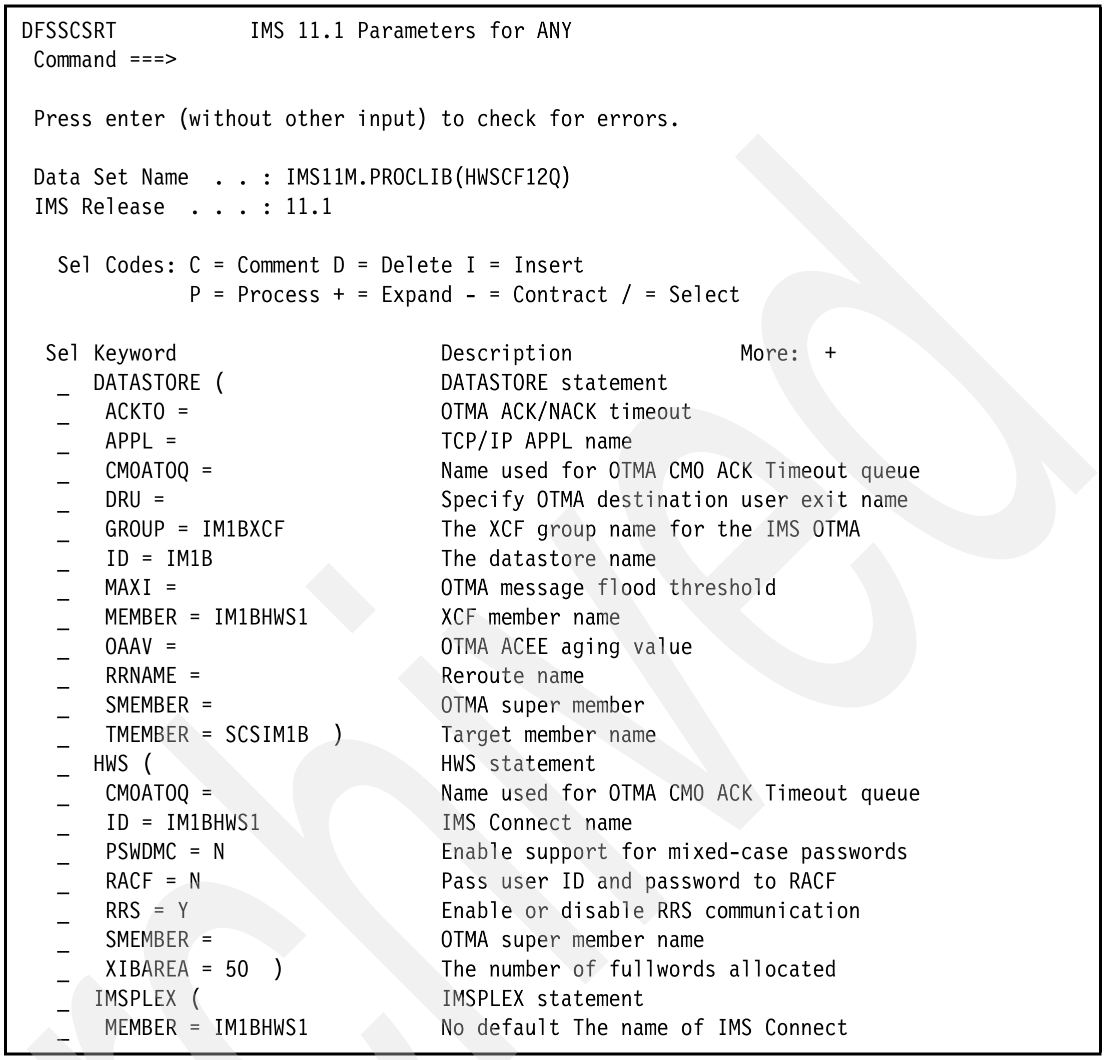

```
 FSSCSRT                 IMS Parameter Syntax Checker
 Command ===>

 Enter the following information and press enter.

 IMS Release  . . . . . . . . 1  1. IMS 11.1
                                 2. IMS 10.1
                                 3. IMS 9.1
```

Figure 6-6 IMS version selection

Select option **1** (**IMS 11.1**), and press Enter. The checker starts the verification and displays the result.

The first page is shown in Figure 6-7.



```
DFSSCSRT           IMS 11.1 Parameters for ANY
 Command ===>

 Press enter (without other input) to check for errors.

 Data Set Name  . . : IMS11M.PROCLIB(HWSCF12Q)
 IMS Release  . . . : 11.1

   Sel Codes: C = Comment D = Delete I = Insert
              P = Process + = Expand - = Contract / = Select

  Sel Keyword                     Description             More:  +
   _  DATASTORE (                 DATASTORE statement
   _   ACKTO =                    OTMA ACK/NACK timeout
   _   APPL =                     TCP/IP APPL name
   _   CMOATOQ =                  Name used for OTMA CMO ACK Timeout queue
   _   DRU =                      Specify OTMA destination user exit name
   _   GROUP = IM1BXCF            The XCF group name for the IMS OTMA
   _   ID = IM1B                  The datastore name
   _   MAXI =                     OTMA message flood threshold
   _   MEMBER = IM1BHWS1          XCF member name
   _   OAAV =                     OTMA ACEE aging value
   _   RRNAME =                   Reroute name
   _   SMEMBER =                  OTMA super member
   _   TMEMBER = SCSIM1B  )       Target member name
   _  HWS (                       HWS statement
   _   CMOATOQ =                  Name used for OTMA CMO ACK Timeout queue
   _   ID = IM1BHWS1              IMS Connect name
   _   PSWDMC = N                 Enable support for mixed-case passwords
   _   RACF = N                   Pass user ID and password to RACF
   _   RRS = Y                    Enable or disable RRS communication
   _   SMEMBER =                  OTMA super member name
   _   XIBAREA = 50  )            The number of fullwords allocated
   _  IMSPLEX (                   IMSPLEX statement
   _   MEMBER = IM1BHWS1          No default The name of IMS Connect
```

Figure 6-7 Page 1 of the Syntax checker result

All fields are verified by the checker, and, with no errors, we can use the member for starting the IMS Connect. Remember that this is a syntax checker; invalid functional values can NOT be verified.

6.1.4 Starting IMS Connect

Example 6-4 shows the JCL to run IMS Connect. In this example, we run IMS Connect under BPE, an option made available with IMS 10. Running by using PGM=HWSHWS00 is still possible. Both IMS Connect and BPE have configuration members (HWSCFG00 and BPECFGHT in this case).

Example 6-4 IMS Connect procedure

```
//HWS       PROC  RGN=4096K,SOUT=A,
//             BPECFG=BPECFGHT,
//             HWSCFG=HWSCFG00
//==================================================================
//* IMS CONNECT USING BPEINI00
//==================================================================
//STEP1 EXEC PGM=BPEINI00,REGION=0M,TIME=1440,
// PARM='BPECFG=&BPECFG,BPEINIT=HWSINI00,
// HWSCFG=&HWSCFG'
//==================================================================
//STEPLIB  DD   DSN=IMS.SDFSRESL,DISP=SHR
//         DD   DSN=CEE.SCEERUN,DISP=SHR
//         DD   DSN=SYS1.CSSLIB,DISP=SHR
//         DD   DSN=GSK.SGSKLOAD,DISP=SHR
//PROCLIB  DD   DSN=USER.PROCLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=&SOUT
//SYSUDUMP DD   SYSOUT=&SOUT
//HWSRCORD DD   DSN=HWSRCDR,DISP=SHR
```

Note: IMS Connect requires the CEE.SCEERUN, SYS1.CSSLIB, and GSK.SGSKLOAD libraries (which are the C execution and z/OS system SSL libraries) only when SSL support is used and these libraries are not link listed.

For more information, read about sample JCL to start IMS Connect information in *IMS Version 11 System Definition,* GC19-2444.

The result of the start can be examined in the z/OS log or in the logging of the IMS Connect task. Example 6-5 shows the messages you might see.

Example 6-5 IMS Connect starting log

```
IEF695I START IM1BCONN WITH JOBNAME IM1BCONN IS ASSIGNED TO USTC     , GROUP SYS1
$HASP373 IM1BCONN STARTED
HWSM0590I CONNECTED TO IMSPLEX=PLEXB   ; M=OSC1
HWSD0290I CONNECTED TO DATASTORE=IM1B    ; M=DSC1
HWSD0290I CONNECTED TO DATASTORE=IM2B    ; M=DSC1
HWSR0653I PROTECTED CONVERSATION PROCESSING WITH RRS/MVS ENABLED M=RRSI
HWSS0790I LISTENING ON PORT=7000     STARTED; M=SDOT
HWSS0780I TCPIP COMMUNICATION ON HOSTNAME=TCPIP    OPENED; M=SOC1
HWSS0790I LISTENING ON PORT=7001     STARTED; M=SDOT
HWSS0790I LISTENING ON PORT=6000     STARTED; M=SDOT
HWSS0790I LISTENING ON PORT=6001     STARTED; M=SDOT
HWSC0010I WELCOME TO IMS CONNECT!
227 HWSC0000I *IMS CONNECT READY*  IM1BHWS1
HWSN1900I IMS CONNECT IS CONNECTED TO ODBM=IM1BOD  ; M=NSC1
HWSN1900I IMS CONNECT IS CONNECTED TO ODBM=IM2BOD  ; M=NSC1
```

Notice the connections to the ODBMs through the SCI interface are established. Through this path, we can access DLI databases directly. The TCP/IP ports for database access are 6000 and 6001. IMS Connect recognized two ODBM members (IM1BOD and IM2BOD) because they were both already started. The connection to control regions IM1B and IM2B through OTMA is also okay. This path is for executing IMS transactions from SOA clients. The TCP/IP ports for transaction access are 7000 and 7001.

No SSL port has been defined.

6.1.5 IMS Enterprise Suite V1.1 Connect API for Java and C

This function is supported on z/OS V1.9 and Windows XP (SP2). Connect APIs extend connectivity of distributed platforms to IMS and simplify application development for stand-alone user-written IMS Connect clients.

The IMS Connect API is intended for all customers who write client applications that interact directly with IMS Connect (such as IMS Connect clients that do *not* use IMS TM Resource Adapter or IMS SOAP Gateway).

To shield client applications (and their developers) from the complexities of interacting with TCP/IP sockets and IMS Connect, the IMS Connect API generates the IMS Connect input message header internally and internally manages the interaction with IMS Connect according to the IMS Connect message protocols. In addition, the IMS Connect API handles socket connections to IMS Connect made on behalf of the client applications including managing pools of sockets if configured to do so.

Figure 6-8 shows how the IMS Connect API hides the complexity of communication protocols for interactions with IMS Connect behind a simple programming interface.

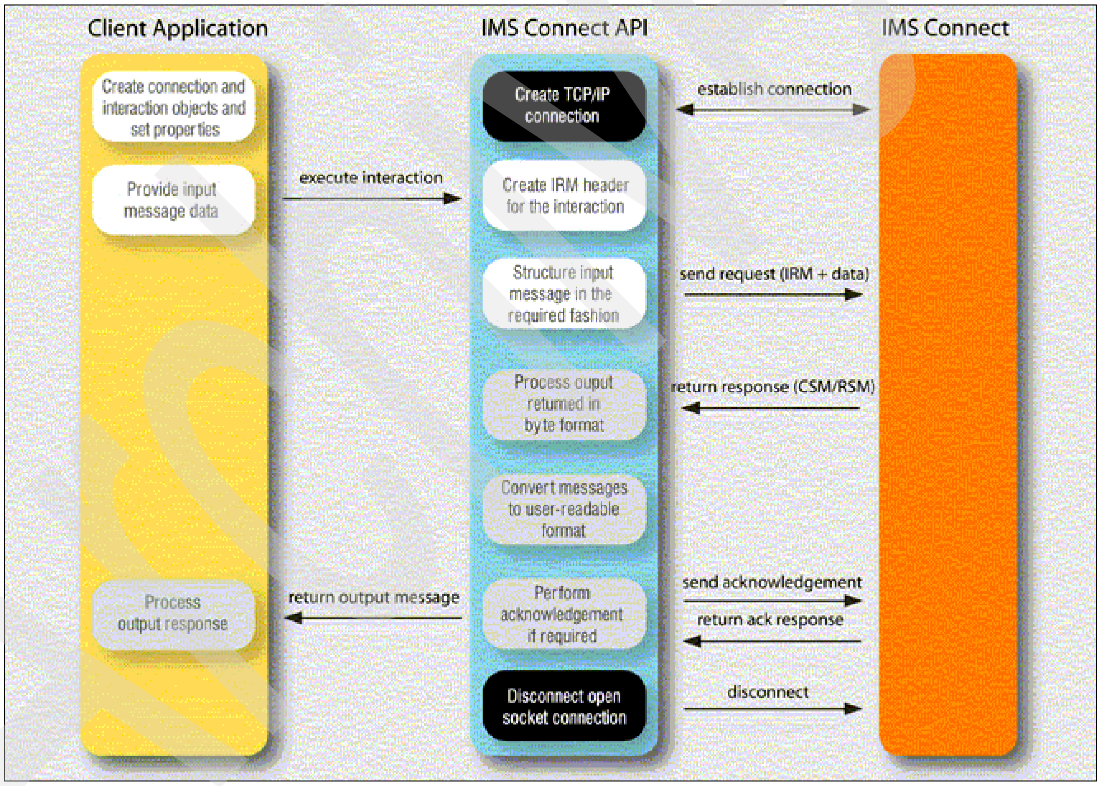

Client Application
IMS Connect API
IMS Connect
Create connection and interaction objects and set properties
Provide input message data
execute interaction
Create TCP/IP connection
establish connection
Create IRM header for the interaction
Structure input message in the required fashion
send request (IRM + data)
Process ouput returned in byte format
return response (CSM/RSM)
Convert messages to user-readable format
Process output response
return output message
Perform acknowledgement if required
send acknowledgement
return ack response
Disconnect open socket connection
disconnect

Figure 6-8 Connect API

The client application sets connection and interaction properties by loading from a profile, or configures the property values programmatically. To perform an interaction, the client application simply issues an execute function call. The interaction types supported by the IMS Connect API include SENDRECV, RESUMETPIPE, and several SENDONLY interaction types.

Internally, the API handles all the work of establishing a connection, creating an IMS Connect input message, sending the input message to IMS Connect, retrieving any output message returned by IMS Connect, and storing the message in a data structure for easy client application retrieval. Behind the scenes, the API also handles any acknowledgement processing required by the IMS Connect protocol, and disconnects from IMS when requested by the client application or when the client application ends.

The API also provides application developers with finer-grain control of connection and interaction handling. For example, developers can: explicitly connect and disconnect to IMS; set the length and trancode for input data; specify settings for the commit mode, synchronization level, different resume-tpipe processing options, and message routing; and dynamically send negative or positive acknowledgement messages.

The IMS Connect API supports profiles for added efficiency and flexibility to IMS customers. Profiles are plain-text files containing property name-value pairs that can be predefined for specific connection and interaction scenarios and reused by different application clients to load connection and interaction property values. Developers can also choose to configure the connection and interaction properties dynamically in the client by using property-setter methods.

The API can be used to issue OTMA-supported IMS commands as well as the IMS Connect PING and RACF password change commands. Data encryption and client/server authentication with SSL is supported by the API. The API also provides support for logging and tracing.

IMS offers both the IMS Connect API and IMS TM Resource Adapter as application development solutions for communicating with IMS through IMS Connect. Table 6-1 highlights the key usage considerations and differences of these two solutions.

Table 6-1 IMS Connect API versus TM resource adapter

Usage consideration	IMS Connect API	IMS TM Resource Adapter
Tools integration	Stand-alone solution that does not require any IBM application development tooling.	Integrated with the IBM Rational® IDEs and WebSphere.
Runtime environment	Best suited for developing IMS Connect RYO client applications.	Best suited for a Java Platform, Enterprise Edition (Java EE) environment.
Deployment method	The JAR file containing the API is deployed on the Java class path; deployment does not require a Java application server environment such as WebSphere.	Deployed and set up to use Java EE Connector Architecture (JCA) connectivity using the administrative console in WebSphere Application Server.
IMS Connect functionality supported	Supports major functions, including CancelTimer; user-defined clientId; and AUTO, NOAUTO, and SINGLE ResumeTpipe interactions which are not supported by TMRA.	Supports all major functions, including two-phase commit global transactions and synchronous call-out, which are not supported in the first release of the IMS Connect API.
Cost	Free offering	Free offering
Connection management	Does not currently share connections across applications.	Uses WebSphere Application Server connection management for client application connection sharing.

Based on these usage considerations, your business requirements, and your application developer's level of knowledge, the IMS Connect API might be the ideal choice for you!

Connect API for Java

Connect API for Java provides programming control of connections to IMS Connect, interactions with IMS, and the data to sent to IMS for those interactions. This set of APIs provide simple interfaces for developing custom IMS Connect TCP/IP client applications.

Connect API for C/C++

IMS Connect API data definition is supplied as a C library header file containing:

- Definitions of all IMS Connect API constants and data types
- Definitions of IMS Connect API C functions
- External C functions for each C++ function

C/C++ Client applications that want to use IMS Connect API must include the library header file and set the path to DLL.

Client application is responsible for managing memory, such as allocating and de-allocating memory for C/C++ data structures.

6.2 IBM IMS Enterprise Suite SOAP Gateway

IBM IMS Enterprise Suite SOAP Gateway, previously a separate product named IMS SOAP Gateway, is a Web service solution that integrates IMS assets in a service-oriented architecture (SOA) environment.

IMS Version 10 was the last release of the separated product. Customers using IMS SOAP Gateway v10 should migrate to IMS Enterprise Suite v1.1.

IBM IMS Enterprise Suite SOAP Gateway is supported on the following platforms

- z/OS V1.9
- Microsoft® Windows XP, Windows 2000
- Linux for System z on SLES9 SP3 64-bit on z/OS
- AIX® V5.3

SOAP Gateway is compliant with the industry standards for Web services, including SOAP/HTTP 1.1 and Web Services Description Language (WSDL) 1.1.

The position of the SOAP Gateway with respect to IMS Connect is shown in Figure 6-9.

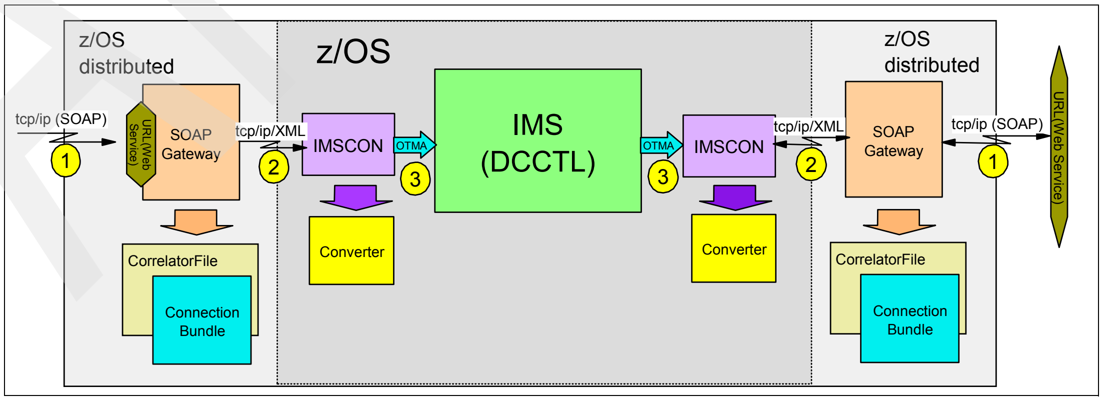


Figure 6-9 Call-in and call-out with SOAP Gateway

IMS SOAP Gateway enables IMS applications to interoperate outside the IMS environment through SOAP to provide and request services independent of platform, environment, application language, or programming model. It assists you to enable your IMS application as a Web service provider and a consumer. Various types of client applications, such as Microsoft.NET, Java, and third-party applications, can submit SOAP requests into IMS to drive the business logic of the COBOL applications. Your IMS application can also make outbound requests to access Web services. As shown in the figure, the follow steps occur:

1. Between Web Service client and SOAP gateway, SOAP format is used over TCP/IP. The frame is composed of the SOAP envelope, which in turn is composed of the SOAP header (mainly used for security) and the SOAP body which contains the data. The complete frame format is XML.

   The SOAP Gateway takes off the particular SOAP (Web Service elements) on call-in, or adds it for the response on call-out.
2. Between SOAP Gateway and IMS Connect the data format is XML over TCP/IP but not SOAP. The XML data are part of an IMS Connect message with the required headers.
3. Conversion from or to ML data is done in IMS Connect. Over OTMA we have a normal IMS format, enhanced with special header but transparent for the application.

The diagram shows two IMS Connects, but only one is sufficient.

6.2.1 Enhancements in IMS Enterprise Suite Version 1 SOAP Gateway

All new features in IMS Enterprise Suite Version 1 SOAP Gateway that are not previously available in IMS SOAP Gateway Version 10 require IMS Version 11 with IMS Connect.

IMS Enterprise Suite Version 1 SOAP Gateway has been partially redesigned. Even though XML message exchange is at the core of Web services, most Web service applications are not concerned with XML. Instead, these applications want to exchange business data that is specific to the application. XML is, in this case, just a format used to represent the business data to support a Web service interface. XML works well for this purpose, because it provides a platform-independent representation that can be handled by a variety of tools. But applications ultimately have to convert the XML to or from their own internal data structures to use the data within the application. *Data binding* is the term used for techniques that handle this conversion between XML and application data structures. XMLBeans, from the Apache Foundation, is a general XML processing framework that includes a data-binding layer. The XMLBeans framework is used by the gateway, and the artifacts generated during the deploying of the Web Services in the gateway server, are prepared to be used also in this environment. In "Java client logic for ODBM access" on page 199, we show the Java code that is required for a client to access the Web Service through the IMS Enterprise Suite Version 1 SOAP Gateway.

Multi-segment message support

Multi-segment messages are not supported for IMS application call-out request messages. Multi-segment messages are supported only when IMS applications are enabled as Web service providers. You can enable multi-segment IMS applications as a Web service provider by clearly identifying the layouts of the input and output messages by using the IBM Rational Developer for System z Version 7.5.1 or later.

For SOAP Gateway and IMS Connect to be able to handle the multi-segment messages, both the input and output message structures must be defined and known ahead of time. By using Rational Developer for System z (RD/z), you can identify the language structures that comprise the input to and output from a multi-segment IMS application from your COBOL source file to generate the XML converter program.

In addition, to identify the language structures that comprise the request and response messages, you must also describe the order and the maximum number of times a pattern can repeat. Just as today for the current single segment handling, the bulk of the multi-segment handling is done by the XML converters that are generated by RD/z and that are deployed onto IMS Connect.

The IMS Connect XML Adapter function utilizes the XML converters for the XML-to-bytes conversion on the way in and bytes-to-XML conversion on the way out of IMS. The multi-segment handling in SOAP Gateway is achieved in two steps:

1. Artifact generation using Rational Developer for System z v7.5.

   An application developer uses the RD/z v7.5 tooling to generate the XML converters, which are subsequently deployed into IMS Connect. In other words, the RD/z tooling is used to support generation of Web services artifacts for multi-segment MPPs.

2. Runtime handling in IMS Connect by the XML Adapter and the generated XML converters.

   At runtime, an incoming multi-segment message is consumed by the XML converters through the IMS XML Adapter function to chop the message into multiple segments that an IMS application could consume. On the way out, the XML converters consolidate the messages from multiple segments into one huge message for IMS SOAP Gateway to consume. There could be multiple inbound or multiple outbound (or multiple inbound and outbound) segments for a request/response message.

Web Service Security

Web Service Security (WS-Security) outlines support for dynamic authentication on a per message basis for accessing IMS applications as Web services. It details message-level security. Message-level security travels with the message.

Transport-level security (for example, HTTPS) is a point-to-point security model where the channel is protected between two parties. However, many times the service consumer and service provider are separated by intermediaries (for example, an enterprise service bus). In situations like these, message-level security can provide an end-to-end security solution. Figure 6-10 shows how message-level security can provide an end-to-end security solution even if intermediaries are between the consumer and provider.

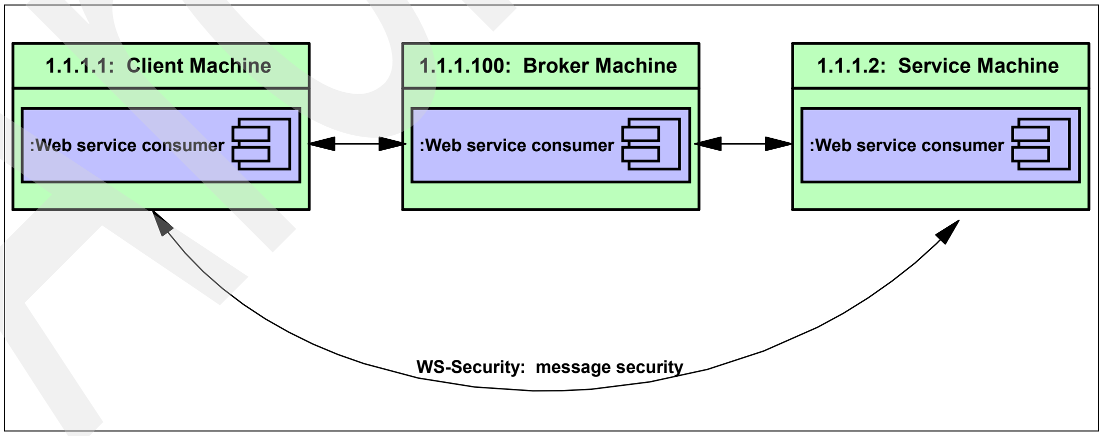


Figure 6-10 Comparison of transport level security and message level security

The secret is that with message-level security, you can encrypt the message using the public key of the final destination.

In this way, only the intended receiver can decrypt the message. Additionally, by encrypting the message and storing the encrypted data into the message, you can store the message on the file system for asynchronous communication and later decrypt it when the receiver is available.

Note: Web Services Security (WS-Security) is an OASIS standard to describe how to implement message-level security with Web services. Specifically, WS-Security describes how you can add confidentiality (for example encryption), integrity (for example digital signatures), and authentication (for example username and password) to a SOAP message by means of:

- Identification and authentication through WSSE tokens
- Username, X509, LTPA
- Integrity through XML Digital Signatures across various parts of message (body, authentication token, time stamp)
- Confidentiality through XML encryption across various parts of message (body, USERNAME token, time stamp)

As a first step, the IMS Enterprise Suite Version 1 SOAP Gateway implements identification/authentication through WSSE tokens. This is an end-to-end authentication based on username/password. The username is in clear text and the password has an elementary encryption based on 64-bit encoding, which is *not* secure. For better security, you must add Transport Layer Security (TLS). This means in practice the usage of cryptography between the hops. Between the hops, we can use HTTPS and SSL.

Figure 6-11 shows that WS-Security is composed of three responsibilities (authorization, confidentiality, integrity). Because the complete suite is not yet implemented, protection of the data has to be complemented by *transport level security*.

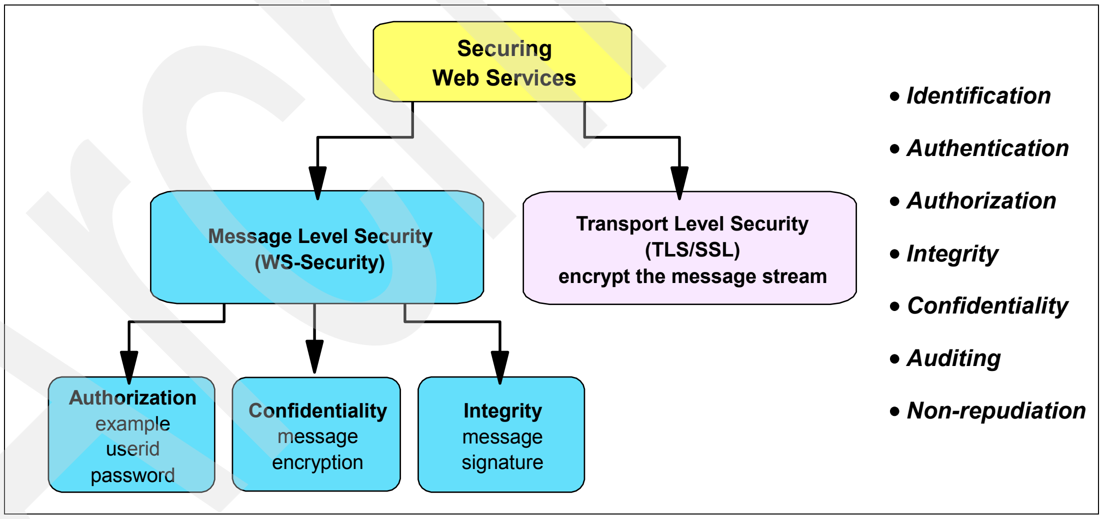

Securing Web Services
Message Level Security (WS-Security)
Transport Level Security (TLS/SSL) encrypt the message stream
Authorization example userid password
Confidentiality message encryption
Integrity message signature
• Identification
• Authentication
• Authorization
• Integrity
• Confidentiality
• Auditing
• Non-repudiation

Figure 6-11 Securing Web Services

SSL and HTTPS security allow data to be transferred in a secure manner between the Web service and IMS SOAP Gateway, and between IMS SOAP Gateway and IMS Connect.

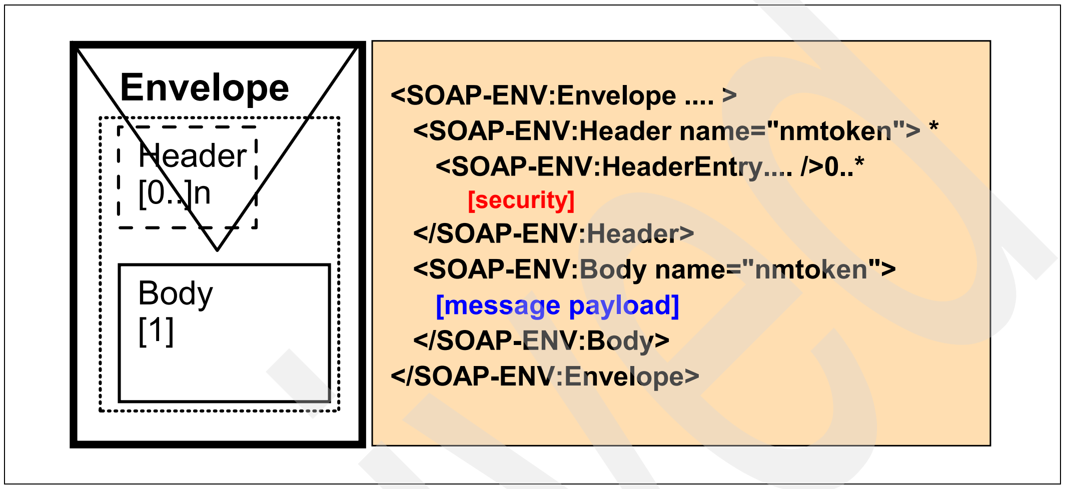
Note the following information:

- The IMS SOAP Gateway communicates with the external WS-service as a consumer, or accepts (as a WS-service provider) requests from external consumers. This communication uses the HTTP(S) request/response protocol. The SOAP frame is imbedded in the HTTP frame.

Building the HTTP(S) frame and the imbedded SOAP layout on call-out is the responsibility of the gateway and includes the build of a correct SOAP envelope with SOAP security header and body.

Stripping off the SOAP headers and passing the XML payload (SOAP Body) to IMS Connect is the responsibility of the gateway on call-in.

Figure 6-12 shows the layout of the SOAP envelope. What is transferred between IMS Connect and the SOAP gateway is the payload, but is in XML format.



```
<SOAP-ENV:Envelope .... >
  <SOAP-ENV:Header name="nmtoken"> *
    <SOAP-ENV:HeaderEntry.... />0..*
       [security]
  </SOAP-ENV:Header>
  <SOAP-ENV:Body name="nmtoken">
    [message payload]
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure 6-12 SOAP envelope

- The IMS SOAP gateway communicates with IMS Connect as a client, and takes place over TCP/IP. On call-out, a tpipe-resume process will solicit messages from IMS Connect as payload for an outgoing Web Service request. On call-in, the payload is a consumer request. The payload is in XML format. SSL can be used to protect this payload transport.

Using WS-security is not mandatory. You can either enable WS-Security feature by the Rational Developer for System z tool or by IMS Enterprise Suite SOAP Gateway deployment utility tool. IBM Rational Developer for System z Version 7.6 is a prerequisite.

Without WS-Security enablement, user identification and password must be provided by connection bundle on a per Web service basis. The security properties between IMS SOAP Gateway, IMS Connect, and IMS are specified in the connection bundle. The security properties that must be specified include:

- User ID
- Password
- Groupname

With enabled WS-Security on call-in, SOAP Gateway extracts user information from the WS-Security header and propagates it to IMS. This extraction is based on the WS *policy set* and its bindings. The policy set indicates the general rules; the binding outlines the practical specification of (usernames, password, keystores). Refer to Example 6-6 on page 185.

Using WS-security has a large influence on the contents of the SOAP envelope, in which mainly SOAP headers show up.

Example 6-6 Excerpt of a WS-security header with basic authentication

```
<soapenv:Envelope xmlns:q0="http://www.IMSPHBKI.com/schemas/IMSPHBKIInterface” ........
   <soapenv:Header xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <wsse:Security soapenv:mustUnderstand=”1" xmlns:wsse=...>
>        <wsse:UsernameToken wsu:Id="user1" ..... >"
            <wsse:Username>user01</wsse:Username>
            <wsse:Password ....>
              password
            </wsse:Password>
         </wsse:UsernameToken>
      </wsse:Security>
      <wsa:To>https://localhost:8443/imssoap/services/IMSPHBKService</wsa:To>"
      <wsa:MessageID>
         urn:uuid:89C2332A42CF7FA1671232146960443
      </wsa:MessageID>
      <wsa:Action>
         urn:IMSPHBK
      </wsa:Action>
   </soapenv:Header>
   <soapenv:Body>
   </soapenv:Body>
</soapenv:Envelope>
```

In addition, certain SSL parameters are required for the IMS SOAP Gateway to communicate with the IMS Connect through TCP/IP SSL sockets. These parameters include:

- SSL keystore name
- SSL keystore password
- SSL truststore name
- SSL truststore password
- SSL encryption level

To specify this data, adapt the connection bundle file. You can access the connection bundle through the IMS SOAP Gateway deployment utility. The userid, group, and password from the connection bundle file are passed to IMS Connect.

If RACF=Y is specified in the HWSCFGxx member, then IMS Connect issues a RACF call to verify these parameters. If the call is not successful, an error message is returned to the IMS SOAP Gateway. If the call is successful, the userid and group is passed to OTMA.

The IMSLSECX exit is also invoked if it exists, but this exit does not verify the password.

Business events support

Business events provide support for monitoring IMS business activities and instrumenting IMS business logic with IBM business processing and monitoring engines such as IBM WebSphere Business Events and IBM WebSphere Business Monitor, it allows IMS applications to participate in business event processing solutions

IMS applications can emit business events to IBM business event engines, such as IBM WebSphere Business Events and IBM WebSphere Business Monitor, for business activities processing and monitoring through SOAP Gateway.

A business event is the representation of a business activity that occurs inside or outside the enterprise or business. An event is a notification to signal a problem, an opportunity, a threshold, or a deviation. The occurrence of the event can trigger activities in another business application, or it can simply trigger data movement.

WebSphere Business Events

WebSphere Business Events is a business event processing engine that supports advanced event processing features for detecting, evaluating, correlating, and responding to events and complex event patterns. To emit business events to WebSphere Business Events, SOAP Gateway leverages the existing SOAP asynchronous call-out capability.

WebSphere Business Monitor

WebSphere Business Monitor is a comprehensive business activity monitoring solution that provides the capability to measure business performance, monitor in-flight and completed processes, and report on business operations. To emit business events to WebSphere Business Monitor, SOAP Gateway asynchronously sends out the event by using the REST protocol. REST provides a lightweight XML API that communicates through HTTP.
IMS applications can emit business events placing the event data on an IMS OTMA hold queue for SOAP Gateway to retrieve and emit to the business event processing engine by using either the SOAP or REST protocol.

Figure 6-13 shows the Business Event Processing Flow. IMS applications can emit business events, placing the event data on an IMS OTMA hold queue for SOAP Gateway to retrieve and emit to the business event processing engine by using either the SOAP or REST protocol.

The business events support in SOAP Gateway works in a similar fashion as the asynchronous call-out function. Business event data is placed in an OTMA hold queue by using the ISRT ALTPCB call in the IMS application to be retrieved by SOAP Gateway. Depending on how the event processing engine expects the event data to be sent, the correlation file that provides information about transaction and runtime properties and about how to match the requests and responses between IMS and the external business event processing engine.

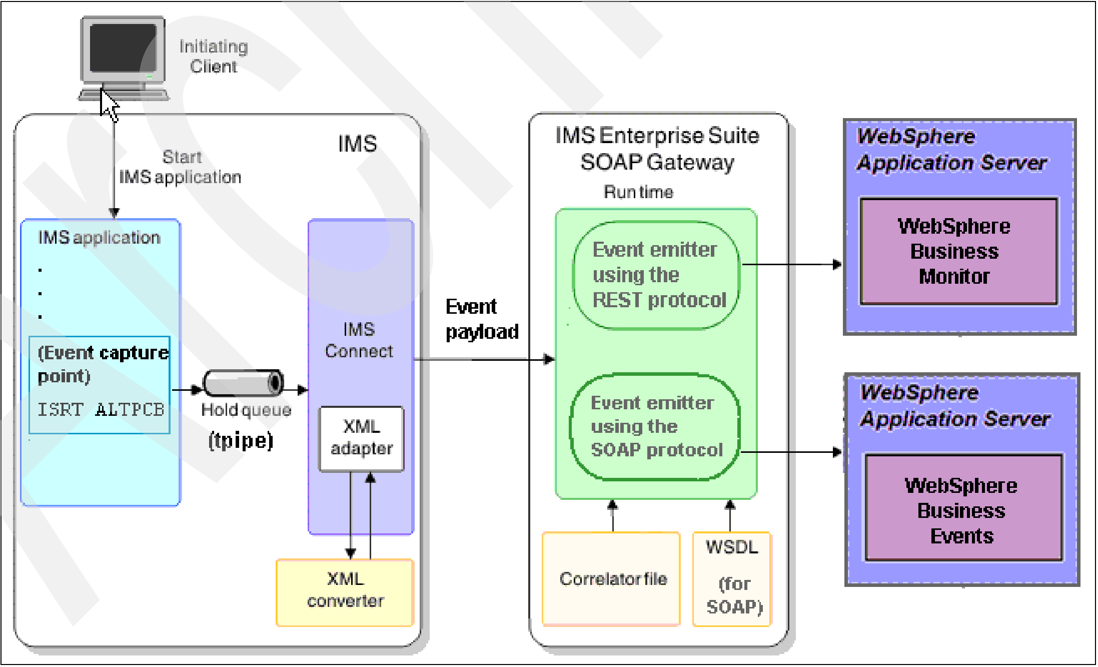

Initiating Client
Start IMS application
IMS
IMS application
.
.
.
(Event capture point)
ISRT ALTPCB
Hold queue
(tpipe)
IMS Connect
XML adapter
XML converter
Event payload
IMS Enterprise Suite SOAP Gateway
Run time
Event emitter using the REST protocol
Event emitter using the SOAP protocol
Correlator file
WSDL (for SOAP)
WebSphere Application Server
WebSphere Business Monitor
WebSphere Application Server
WebSphere Business Events

Figure 6-13 Business Event Processing Flow

The process flow is as follows:

1. An IMS application emits business event data by issuing an ISRT ALTPCB call to an OTMA destination descriptor, which contains the destination tpipe name. The message that contains the business event data is queued in this tpipe.
2. The IMS Connect XML adapter function converts the business event data from bytes to XML by using the XML converter that is generated by Rational Developer for System z from the IMS application source file.
3. SOAP Gateway retrieves the business event message from the tpipe. Based on the correlator file that is generated by Rational Developer for System z, SOAP Gateway emits the business event by using either the SOAP protocol or the REST protocol.

6.3 IBM IMS Enterprise Suite DLIModel utility

DB2 databases have a catalog with metadata; DLI databases do not. The IBM IMS DLIModel utility allows you to transform your IMS database information into application-independent metadata to use for application development. The base for these metadata are the PSB and associated DBDs. In general the field information in the DBDs is not very elaborated, often limited to the keyfield. The Utility allows the adding of more field information from copybooks/includes, manually, or both.

Two versions of the DLIModel utility are available:

- An included IMS version that runs from UNIX System Services or from the z/OS BPXBATCH utility
- A Web download version that runs as a plug-in to Eclipse, WebSphere Developer for System z, Rational Application Developer Software for WebSphere, and Rational Developer for System z.

Attention: IMS Version 10 is the last release of IMS to support the version of the DLIModel utility included in IMS. Customers using this version should migrate to using the DLIModel utility plug-in (Eclipse plug-in).

6.3.1 IMS Enterprise Suite DLIModel utility plug-in

The IMS Enterprise Suite DLIModel utility plug-in translates IMS source files into reliable metadata that can be used for Java application development in an Eclipse-based environment, eliminating the need for hand-coding.

In order for a Java application to access an IMS database, it needs information about that database. This database information is contained in IMS source files, such as PSBs (program specification blocks) and DBDs (database descriptions). To access this information, you must first convert it into a form that you can use in the Java application: a subclass of the `com.ibm.ims.db.DLIDatabaseView` class that is called the IMS Java metadata class. The DLIModel utility generates this metadata from IMS source files.

If you have an existing IMS control statement to use with the version of the DLIModel utility that is included with IMS, you can reuse your control statement as a source for generating metadata instead of individually specifying PSB and DBD files.

This metadata is in the form of a Java class, `DLIDatabaseView` class, prepared with the DLIModel utility plug-in under Rational Application Developer.

Enhancements in IMS Enterprise Suite DLIModel utility plug-in

The plug-in, which can be installed on Rational Application Developer V7.5, has the following additional features compared to its previous version:

- PSB/PCB *procopt* levels are already verified in the client at runtime.

  For example, if `PROCOPT=GO` is specified in the PSB, the corresponding DLIDatabaseView does not allow an SQL `INSERT` call. Previously, this control took place in IMS, and then a reject-code had to be returned all the way back.

- Cobol Copybooks and PLI include statements can be used to complete the DLIDatabaseView class. PSBs and DBAs are not altered.
- The plug-in, after selection of the PSB, automatically selects the associated DBDs.

  The DLImodel takes as input a PSB, which references DBDs. The PSB and DBD names should match the filenames they reside in. Existing PSBs can be used as is; removing the TPpcbs is *not* required.

- DBD hierarchical structures can be exported to graphic files (JPG or BMP).
- A Virtual Foreign Key view can be added to the current PSB graphical editor.
- A search capability to the current PSB graphical editor is available.
- The existing metadata in a DLIModel utility project can be updated when PSB or DBD sources are modified.

The existing V10 DLIModel plug-in can be downloaded as part of the IMS SOA Integration Suite from:

http://www.ibm.com/software/data/ims/toolkit/dlimodelutility/

It can also be obtained later as part of the IMS Enterprise Suite.

A simple DLIModel example

We prepare a simple DLImodel, which uses the IVPDB1(D21PART) database. Later, we use the result of this example when we explain the IMS 11 capability to access DLI databases without a transaction manager. The DLIModel plug-in utility was first installed under Rational Application Developer.

Example 6-7 shows the PSB with one PCB, which relates to one DBD (IVPDB1).

Example 6-7 PSB DFSIVP residing in file DFSIVP

```
PCB01     PCB    TYPE=DB,DBDNAME=IVPDB1,PROCOPT=A,KEYLEN=10
          SENSEG NAME=A1111111,PARENT=0,PROCOPT=A
          PSBGEN LANG=ASSEM,PSBNAME=DFSIVP
          END
```

Example 6-8 shows the DBD, with a LCHILD statement pointing to a secondary index database IVPDBI1.

Example 6-8 DBD IVPDB1 residing in file IVPDB1

```
        DBD     NAME=IVPDB1,ACCESS=(HIDAM,OSAM)
        DATASET DD1=DFSIVD1,DEVICE=3380,SIZE=2048
        SEGM    NAME=A1111111,PARENT=0,BYTES=40,RULES=(LLV,LAST),PTR=(TB,CTR)
        FIELD   NAME=(A1111111,SEQ,U),BYTES=010,START=00001,TYPE=C
        LCHILD  NAME=(A1,IVPDB1I),POINTER=INDX,RULES=LAST
        DBDGEN
        FINISH
        END
```

Example 6-9 shows the index database.

Example 6-9 DBD IVPDB1I residing in file IVPDB1I

```
       DBD    NAME=IVPDB1I,ACCESS=(INDEX,VSAM,PROT)
       DATASET DD1=DFSIVD1I,DEVICE=3380,SIZE=2048
       SEGM   NAME=A1,PARENT=0,BYTES=10
       FIELD  NAME=(A1,SEQ,U),BYTES=010,START=00001,TYPE=C
       LCHILD NAME=(A1111111,IVPDB1),INDEX=A1111111
       DBDGEN
       FINISH
       END
```

The DLImodel utility generates two items for this PSB:

- DFSIVPDatabaseView, class extension of DLIDatabaseView, which is shown in Example 6-10
- Graphical model `DFSIVP.mdl`, which has several options to enrich the existing DLIDatabaseView class

Note: The DLIModel utility can generate a trace file named `dlimodeltrace` if you need to resolve a problem with the utility.

Example 6-10 DFSIVPDatabaseView

```
package dfsivp;

import com.ibm.ims.db.*;
import com.ibm.ims.base.*;

public class DFSIVPDatabaseView extends DLIDatabaseView {

    // This class describes the data view of PSB: DFSIVP
    // PSB DFSIVP has database PCBs with 8-char PCBNAME or label:
    //       PCB01

    // The following describes Segment: A1111111 ("A1111111") in PCB: PCB01 ("PCB01")
    static DLITypeInfo[] PCB01A1111111Array= {
        new DLITypeInfo("A1111111", DLITypeInfo.CHAR, 1, 10, "A1111111",
DLITypeInfo.UNIQUE_KEY)
    };
    static DLISegment PCB01A1111111Segment= new DLISegment
        ("A1111111","A1111111",PCB01A1111111Array,40);

    // An array of DLISegmentInfo objects follows to describe the view for PCB: PCB01
("PCB01")
    static DLISegmentInfo[] PCB01array = {
        new DLISegmentInfo(PCB01A1111111Segment,DLIDatabaseView.ROOT)
    };

    // Constructor
    public DFSIVPDatabaseView() {
        super("2.0.3","DFSIVP", "PCB01", "PCB01", PCB01array, "A");
    } // end DFSIVPDatabaseView constructor

} // end DFSIVPDatabaseView class definition
```

The current content of the DFSIVPDatabaseView misses the details about the fields in the Database segments. This is required to have any formatted result from DLI or SQL.

With the Import Copybook Fields option, we can bring in additional details. See Figure 6-14. Copybook has to be interpreted as both Cobol Copybooks and PL/I includes.

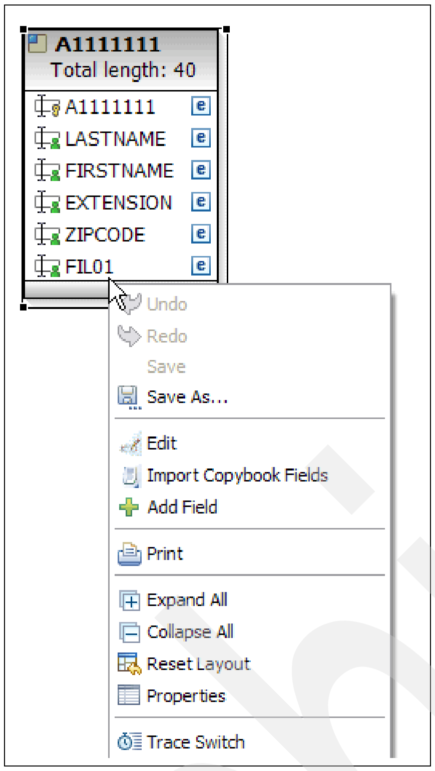


Figure 6-14 Graphical model DFSIVP.mdl with options

The Cobol copybook used has the contents shown in Example 6-11. Cobol copybooks have to be syntactically correct (for example use the right column margins and refrain from using reserved words). This is also true for PL/I includes.

Example 6-11 Contents of the Cobol copybook

```
01  A1111111.
           05  LASTNAME       PICTURE X(10).
           05  FIRSTNAME      PICTURE X(10).
           05  EXTENSION      PICTURE X(10).
           05  ZIPCODE        PICTURE X(7).
           05  FIL01          PICTURE X(3).
```

Finally, after import, the `DLIDatabaseView` class looks similar to Example 6-12 on page 191. The change of the details in the `view` class does not change PSBs or DBDs.

Note: When a field is defined by the DBD, the default data type for all fields is CHAR. When a field is imported from a COBOL or PL/I copybook, the data type is defined by the COBOL or PL/I copybook. You can change the data type of a field by modifying a field.

Fields can also be added manually.

Example 6-12 Final DFSIVPDBView

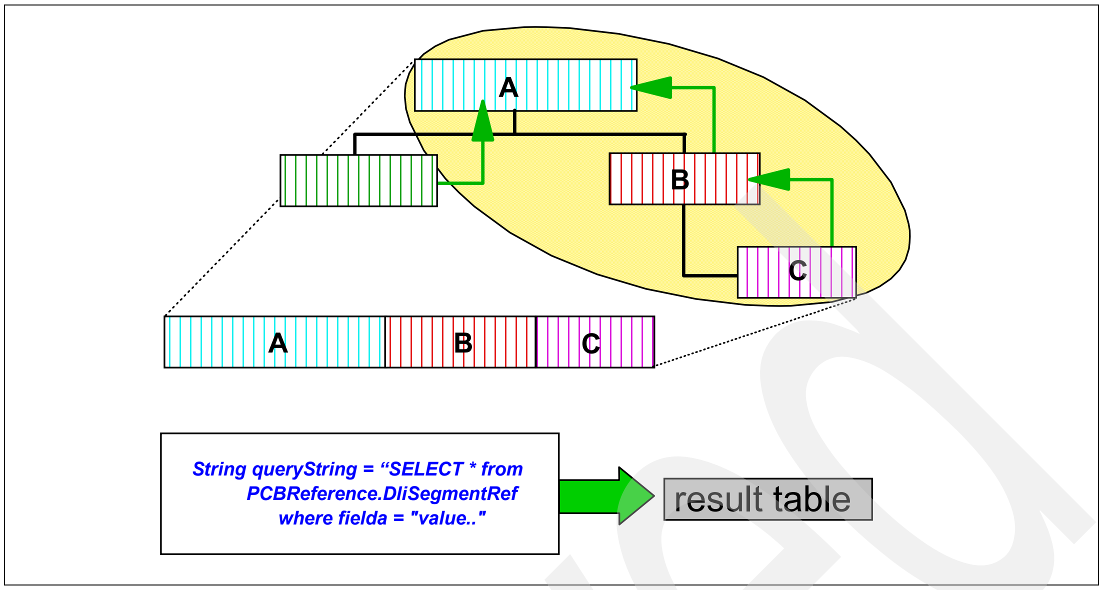
```
package opendb;
import com.ibm.ims.db.*;
import com.ibm.ims.base.*;
/**
 * This class was generated using the DL/I Model Utility using the PSB source DFSIVP1 which generates PSB
DFSIVP1,
 * and DBD source IVPDB1.
 *
 * The database is loaded using DFSIVC04
 */
public class DFSIVPDBView extends DLIDatabaseView {
   // This class describes the data view of PSB: DFSIVP1
   // PSB DFSIVP1 has database PCBs with 8-char PCBNAME or label: PCB01
   // The following describes Segment: A1111111 ("A1111111") in PCB: PCB01 ("PCB01")
    static DLITypeInfo[] PCB01A1111111Array= {
        new DLITypeInfo("LASTNAME", DLITypeInfo.CHAR, 1, 10, "A1111111", DLITypeInfo.UNIQUE_KEY),
        new DLITypeInfo("FIRSTNAME", DLITypeInfo.CHAR, 11, 10),
        new DLITypeInfo("EXTENSION", DLITypeInfo.CHAR, 21, 10),
        new DLITypeInfo("ZIPCODE", DLITypeInfo.CHAR, 31, 7)
    };
    static DLISegment PCB01A1111111Segment= new DLISegment
        ("PhoneBook","A1111111",PCB01A1111111Array,40);
   // An array of DLISegmentInfo objects follows to describe the view for PCB: PCB01 ("PCB01")
    static DLISegmentInfo[] PCB01array = {
        new DLISegmentInfo(PCB01A1111111Segment,DLIDatabaseView.ROOT)
    };
   // Constructor
    public DFSIVPDBView() {
        super("2.0","DFSIVP1", "PCB01", "PCB01", PCB01array);
        PCB01A1111111Segment.setDefaultEncoding("Cp1047");

    } // end DFSIVP1DatabaseView constructor
}  // end DFSIVP1DatabaseView class definition
```

Usage of the DLIDatabaseView class

The DLIDatabaseView class is used by the *adapters* for direct access to DLI databases. The class in this case has to be considered as the *Data source* or the target of the connection.

The class is also used by the Java-dependant regions: Java Batch Processing (JBP) and Java Message Processing (JMP)

The metadata in the DLIDatabaseView class compensate for the lack of information, compared to the row or field information present in the catalog for DB2 table or rows and allows for presenting the DLI data in a relational row format, by concatenating the segments on lower levels with the parents in their hierarchical path. See Figure 6-15 on page 192.


A
B
C
A
B
C
String queryString = "SELECT * from
PCBReference.DliSegmentRef
where fielda = "value.."
result table
Figure 6-15 Concatenation of hierarchical DLI segments

This approach opens the door for SQL access. Each segment in Figure 6-15 has a key. When data is requested on a dependant segment (for example, C), the data is picked up as though A/B/C were concatenated and form only one relational row.

Restrictions for the DLIModel utility plug-in

You cannot use the DLIModel utility plug-in to modify a data structure in IMS on the mainframe. You must acquire source files from z/OS system by copying them over to a distributed platform or using FTP from the mainframe.

The DLIModel utility cannot process:

- MSDB, HSAM, and SHSAM databases
- Shared secondary indexes
- Processing option to make an application sensitive to only the segment key of a segment (PROCOPT=K option) in a PSB SENSEG statement

6.4 IBM IMS Enterprise Suite JMS API

The Java Message Service (JMS) API expands Java application development in Java-dependent regions. JMS API is intended for z/OS V1.9, as a minimum.

Java message processing (JMP) or Java batch processing (JBP) applications can issue synchronous call-out requests to external services through an IMS implementation of the Java Message Service (JMS). To use the JMP and JBP support for synchronous call-out, the JMS version 1.1 interface classes are required. This JMS API is included in the suite for your convenience. IMS provides support for synchronous call-out functionality from Java message processing (JMP) or Java batch processing (JBP) applications through an IMS implementation of the Java Message Service (JMS).

The IMS implementation of JMS is limited to supporting the point-to-point (PTP) messaging domain only. In addition, support is provided only for non-transacted QueueSession objects with `Session.AUTO_ACKNOWLEDGE` mode. If the JMP or JBP application attempts to call any JMS method not supported by IMS or pass any unsupported argument to JMS method calls, a JMSException exception is thrown.

In the IBM IMS Enterprise Suite - Java API specification, package `com.ibm.ims.jms` provides IMS-specific extensions for performing synchronous call-out from JMP or JBP regions. It contains the class IMSQueueConnectionFactory, which is the Factory class used to create IMS point-to-point messaging domain QueueConnection objects. The code fragment in Example 6-13 shows how to use IMSQueueConnectionFactory to obtain and use a QueueConnection object.

Example 6-13 Code excerpt, showing usage of IMS JMSAPI

```
import com.ibm.ims.jms.IMSQueueConnectionFactory;
public class IMS_Sample
{
    public static void main(String argv[])
    {
        IMSQueueConnectionFactory jmsConnectionFactory
           = new IMSQueueConnectionFactory();
        QueueConnection jmsConnection = null;
        QueueSession jmsQueueSession = null;
        Queue jmsQueue = null;
        QueueRequestor jmsQueueRequestor = null;
        try {
            jmsConnectionFactory.setTimeout(1000);
                          // set the timeout to 10 seconds
            jmsConnectionFactory.setResponseAreaLength(128000);
                          // allocate 128k to hold the response message
            jmsConnection = jmsConnectionFactory.createQueueConnection();
            jmsQueueSession
            = jmsConnection.createQueueSession(false, Session.AUTO_ACKNOWLEDGE);
                   // set session to be non-transacted and in AUTO_ACKNOWLEDGE mode
            jmsQueue = jmsQueueSession.createQueue("OTMDEST1");
                   // pass in the OTMA descriptor name

            jmsQueueRequestor
               = new QueueRequestor(jmsQueueSession, jmsQueue);
            TextMessage sendMsg = jmsQueueSession.createTextMessage();
            sendMsg.setText("MyMessage");
            System.out.println("Sending message: "+sendMsg.getText());
            TextMessage replyMsg
               = (TextMessage)jmsQueueRequestor.request(sendMsg);

            System.out.println("\nReceived message: "+replyMsg.getText());
        } catch (JMSException e) {
            e.printStackTrace();
        }
    }
}
```

6.5 IMS as a database (DLI) server in SOA

IMS Connect gave distributed and local programs the opportunity to access existing and new message-processing programs. The development of the connector part for those client programs was rather easy based on the Java Connector Architecture (JCA 1.5 latest version) and with the help of development wizards, in Rational Application Development.

With IMS 11 IMS Connect gets an additional role as gateway to IMS Data Base Control DBCTL, without using transactions. This allows a new easy direct access path to the DLI databases.

6.5.1 Accessing DLI databases in IMS

IMS V10 offered the possibility to access the DLI databases under the DBCTL component of IMS, by using the ODBA capabilities directly. Possibilities also included having JDBC access from Java programs running in batch, or program devices as an artifact, such as Enterprise JavaBeans (EJB) servlet, and deployed in the WebSphere Application Server. In the latter case, a requirements was to install a Resource Adapter in WebSphere Application Server. Both local and remote access were possible. As a reminder, Figure 6-16 shows how this access could be done from a distributed site or locally to IMS.

From the distributed platform to the local z/OS, the IIOP protocol is used to call the *passthru* EJB in the local WebSphere Application Server, which in turn does then the ODBA access.

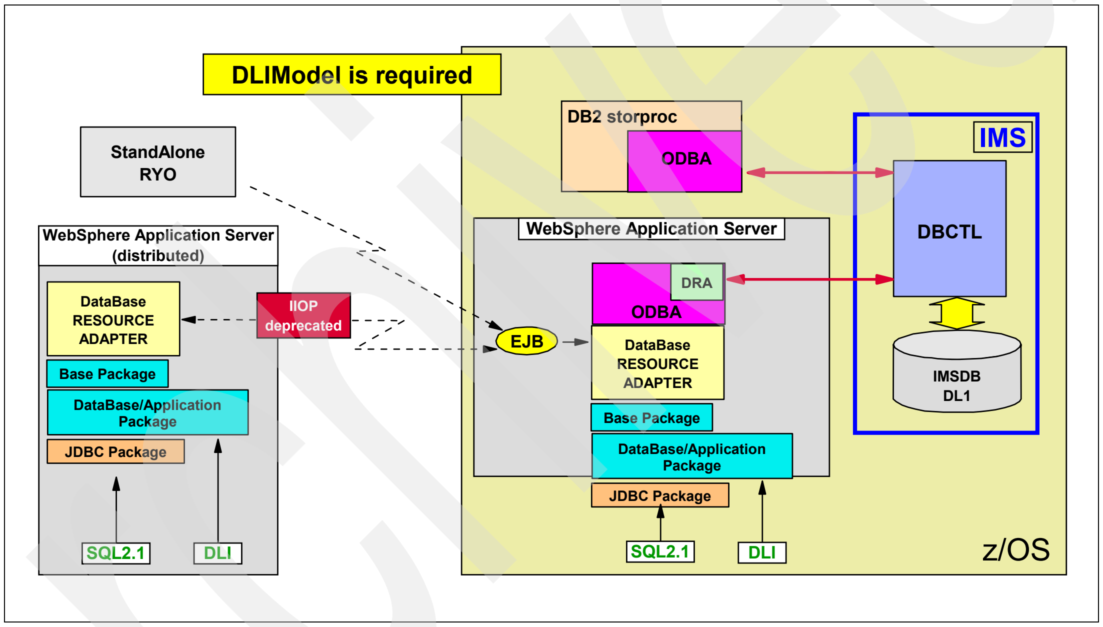

DLIModel is required
StandAlone RYO
DB2 storproc
ODBA
IMS
DBCTL
WebSphere Application Server (distributed)
DataBase RESOURCE ADAPTER
IIOP deprecated
Base Package
DataBase/Application Package
JDBC Package
SQL2.1
DLI
WebSphere Application Server
DRA
ODBA
EJB
DataBase RESOURCE ADAPTER
Base Package
DataBase/Application Package
JDBC Package
SQL2.1
DLI
IMSDB DL1
z/OS

Figure 6-16 Remote and local access to DLI data without ODBM

The IMS 11 access capability offers an important improvement, especially for when the client is remotely (distributed) connected to IMS in z/OS. This capability, which is available from J2EE and non-J2EE environments, is achieved by using a new Open Database Manager (ODBM) address space, which uses ODBA to access DLI data. ODBM runs in a Common Service Layer (CSL) address space and is an optional IMSplex component. ODBM uses the Structured Call Interface (SCI) services of the CSL for communication and Operations Manager (OM) services of the CSL for command processing.

One ODBM instance must be defined in the IMSplex to use ODBM functions. Each z/OS image can have more than one ODBM. If multiple instances of ODBM are defined in the IMSplex, any ODBM instance can perform work from any z/OS image in the IMSplex.

The ODBM address spaces is approached from IMS Connect over a PC[2]/XCF[3] mechanism, which allows communication across LPARs within an IMSPLEX. IMS Connect has dedicated TCP/IP ports, through which a DRDA related protocol is used by clients.

[2] Cross-address-space program call

Figure 6-17 shows how remote and local clients can access DLI databases through ODBM. In practice, those client programs are written in Java, and can fall into two categories:

- Stand-alone non-J2EE programs
- Code that is part of a J2EE artifact in a WebSphere Application Server

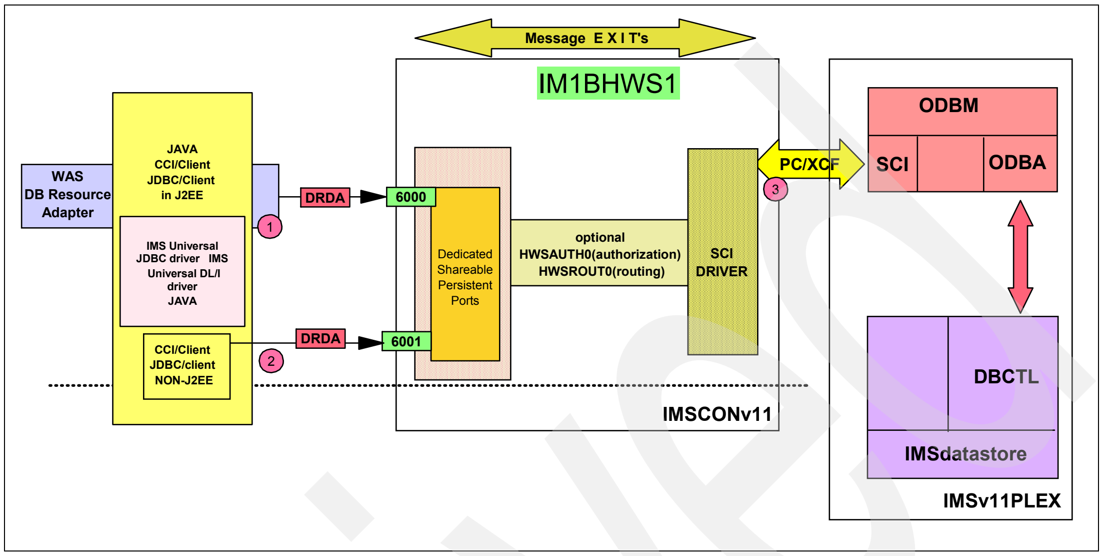

Message E X I T's
IM1BHWS1
WAS DB Resource Adapter
JAVA CCI/Client JDBC/Client in J2EE
IMS Universal JDBC driver IMS Universal DL/I driver JAVA
CCI/Client JDBC/client NON-J2EE
DRDA
DRDA
6000
6001
1
2
3
Dedicated Shareable Persistent Ports
optional HWSAUTH0(authorization) HWSROUT0(routing)
SCI DRIVER
PC/XCF
IMSCONv11
ODBM
SCI
ODBA
DBCTL
IMSdatastore
IMSv11PLEX

Figure 6-17 Access to ODBM through IMS Connect

Both solutions, IMS 10 and 11, have something in common. The metadata for each PSB with DLI access (has a database PCB) that we want to use must be available. That metadata which is in the format of a Java class DLIDatabaseView, are prepared by DLIModel utility plug-in, which is explained in 6.3.1, “IMS Enterprise Suite DLIModel utility plug-in” on page 187.

Open Database Manager (ODBM) provides distributed and local access to IMS databases that are managed by IMS DB systems configured for either the DBCTL or the DB/TM environments in an IMSplex.

Both independently and with IMS Connect, ODBM supports a variety of interfaces to ease the development of application programs that access IMS databases from many different distributed and local environments. The interfaces that ODBM supports include:

- IMS Universal DB resource adapter
- IMS Universal JDBC driver
- IMS Universal DL/I driver
- The ODBA interface
- The ODBM CSLDMI interface

In addition to supporting these interfaces, ODBM provides the following functions:

- Routes incoming database requests to IMS systems based on alias names. Alias names for IMS systems are defined to ODBM on the CSLDCxxx PROCLIB member and specified on incoming database requests by the client application programs.
- If the client application program does not specify an alias name on a database access request, ODBM routes the request among multiple IMS systems by using a round-robin method of distribution, in which ODBM routes each incoming request to the active IMS systems that are defined to ODBM.

[3] Cross-system coupling facility

- Enables ODBM client application programs to access databases from other logical partitions within an IMSplex.
- Protects IMS control regions from the unexpected termination during DL/I processing of z/OS application programs that use the ODBA interface through ODBM.

DRDA for DLI access

IMS 11 provides an implementation of the Distributed Relational Database Architecture (DRDA) protocol that you can use to write roll-your-own (RYO) applications to communicate with IMS Connect over TCP/IP.

DRDA is an open architecture that enables communication between applications and database systems on disparate platforms. Details about using the DRDA protocol to perform database access operations are in the open specifications for DRDA. The IMS support for DRDA is based on the DRDA Version 4 technical standard. The DRDA specification is documented by The Open Group consortium at:

http://www.opengroup.org

To use the IMS support for DRDA, you must create the DRDA client driver (DRDA source server). No additional software has to be installed or configured on the client system. The DRDA target server consists of IMS Connect and the Open Database Manager (ODBM) running with IMS in z/OS.

The IMS support for DRDA includes support for both application-directed transaction demarcation (local) and XA-enabled (global) transactions.

IMS does *not* support the following DRDA functions:

- Multi-row input
- Client reroute
- Security plug-in

The IMS solutions for Java development allow you to write Java application programs that work with IMS. IMS 11 delivers many enhancements to this. Java application programs can access databases that are managed by IMS DB by using any one of three new IMS Universal drivers:

- The IMS Universal DB resource adapter - a JCA-compliant API that provides all the services that the J2EE platform provides, including connection, transaction, and security management
- The IMS Universal JDBC driver - a JDBC driver that supports access to IMS data by using SQL calls
- The IMS Universal DL/I driver - an IMS-specific Java application programming interface (API) for DL/I that can access IMS data by using Java methods that are based on IMS DL/I semantics

Resource Adapters

A resource adapter is a J2EE component that implements the J2EE Connector architecture for a specific enterprise information system (EIS). Through the resource adapter, a J2EE application communicates with an EIS. Depending on their level, resource adapters can be compliant with several specifications, the latest being the Java Connection Architecture (JCA) V1.5.

Resource adapters must be installed in the WebSphere Application Server, they are distributed as Resource Archive (RAR) files. The RAR files do exist for both WebSphere Application Server on z/OS and distributed. Under the installed RAR, with WebSphere

Application Server administration (Admin Console, scripting), preconfiguring ConnectionFactories or Datasources is possible. In this case we mention them as managed resources, which can be looked up from the name space. The required configuration fields are outlined by XML files, which are also part of the RAR.

For IMS 11, we distinguish three categories of adapters:

- Adapters for DLI Access (new in IMS 11)

  Depending on how you want to access the DLI database, some or all resource adapters must be installed in WebSphere Application Server. See Figure 6-18.

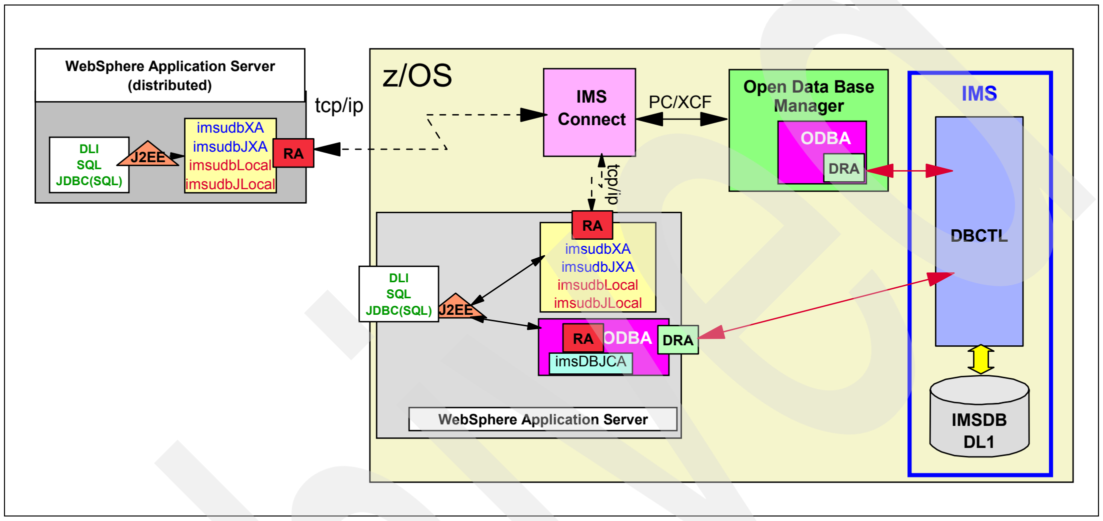


Figure 6-18 RARs should be installed in WebSphere Application Server

> **Note:** If Rational Application Developer is used as development tool, then it is also useful to import the RARs in the workbench, so that during Java programming the IMS/DLI and IMS/JDBC can be addressed. The classes are package in Java Archives (JARs).
>
> The import of the RAR creates a dedicated project that contains all required classes (in JARs). A client project can reference the RAR project. This project must also have access to the *javax CCI* classes if CCI the selected API.

The following RARs have been made available with IMS 11 for accessing the DB component with ODBM through IMS Connect:

– `imsudbLocal.rar`

  Use the IMS Universal DB resource adapter with local transaction support and make SQL calls with the SQLInteractionSpec class or DL/I calls with the DLIInteractionSpec class.

  CCI programming interface should be used to perform SQL or DL/I data operations.

– `imsudbXA.rar`

  Use the IMS Universal DB resource adapter with XA transaction support and make SQL calls with the SQLInteractionSpec class or DL/I calls with the DLIInteractionSpec class.

  This requires two-phase (XA) commit processing or local transaction processing. CCI programming interface should be used to perform SQL or DL/I data operations

– `imsudbJLocal.rar`

  Use the IMS Universal JCA/JDBC driver version of the IMS Universal DB resource adapter with local transaction support, and make SQL calls with the JDBC API.

  JDBC programming interface should be used to perform SQL data operations.

– `imsudbJXA.rar`

  Use the IMS Universal JCA/JDBC driver version of the IMS Universal DB resource adapter with XA transaction support (`imsudbJXA.rar`), and make SQL calls with the JDBC API.

  This requires two-phase (XA) commit processing or local transaction processing. JDBC programming interface should be used to perform SQL data operations.

Tip: for JDBC the (J) drivers are required, XA (2 Phase Commit) requires a (XA) driver

- Adapter for DLI Access (from IMS 10)

  – `imsDBJCA.rar`

Note: Besides the new universal drivers, the older drivers still exists. The distributed solution, requiring a Was Application Server with a deployed EJB should be considered to be deprecated. See Figure 6-16 on page 194.

- Transaction Manager Resource Adapter (TMRA)

  The IMS TM Resource Adapter runs under WebSphere Application Server in either z/OS or distributed environments and is used by Java applications, Java Platform, Enterprise Edition (Java EE, previously known as J2EE) applications or Web services to access IMS transactions that are running on host IMS systems. It is also used by IMS applications that run in IMS dependent regions to make asynchronous call-out requests to external Java EE applications.

  When running in a z/OS environment, the IMS TM Resource Adapter can connect to IMS Connect through either the TCP/IP communications protocol or through z/OS program calls. The use of z/OS program calls is referred to as the local option. When running in a distributed environment, the IMS TM Resource Adapter connects to IMS Connect through only the TCP/IP communications protocol.

  In IMS Version 11, the IMS TM Resource Adapter has two main enhancements.

  – Take advantage of a new function in IMS Connect, the Generated Client ID function, which ensures the uniqueness of each socket.

    This enhancement eliminates the requirement for the IMS TM Resource Adapter to use different IMS Connect ports for instances of distributed WebSphere Application Server. IMS Connect can use the Generated Client ID function to ensure the uniqueness of each socket and allow all instances of WebSphere Application Server to specify the same IMS Connect TCP/IP port.

  – Add Message Format Services (MFS) Business Process Execution Language (BPEL) support.

    This enhancement provides Service Oriented Architecture (SOA) support in WebSphere Integration Developer for IMS Message Format Service (MFS) applications. IMS TM Resource Adapter will have support for IMS MFS Business Process Choreography.

Attention: This enhancement is delivered through the service process.

With this enhancement, you can:

- Transform existing conversational and non-conversational MFS-based IMS applications into components (services) that can be used to create a business process.
- Create business processes involving IMS MFS Import component, which can become components to a bigger business process, with easy-to-use visual tools that comply with the industry-standard Business Process Execution Language (BPEL).
- Enable business integration developers to assemble business solutions involving their existing IMS MFS applications.

IMS MFS support in WebSphere Integration Developer supports Service Component Architecture (SCA), which is a model that provides a simple and consistent means for expressing business logic and business data as SOA services, regardless of implementation details. IMS MFS applications, components, and services created in WebSphere Integration Developer can be deployed to run in WebSphere Process Server.

Note: When ConnectionFactories and Datasources have been preconfigured under their resource adapter, they can be looked up with the Java Name Directory Interface (JNDI) and used directly to get a connection (object). This approach is only possible for J2EE artifacts, otherwise the objects must be built and then populated with the required properties.

For non-J2EE logic, building the objects and then populating them with the required properties is the most appropriate solution, because of the difficulty in accessing a *name space* directory

Java client logic for ODBM access

Mostly client logic, using ODBM access to DLI databases, is written in Java. The IMS DB resource adapters provide the interface to IMS DB in a J2EE environment and implements several APIs (DLI, SQL, JDBC, which is a much fuller implementation than in previous IMS releases). As a result, the programs can be written in several variations:

- ConnectionSpec connection and PSB handler
- JC1 1.5 ConnectionFactory connection with either SQL or DL Interaction
- DataSource connection and JDBC access

Using IMSConnectionSpec (not CCI) with PSB handler

Refer to Example 6-14.

Example 6-14 Connection with IMSConnectionSpec

```
// Create an IMS Connection Specification which will allow you to access your IMS data
     IMSConnectionSpec connSpec = IMSConnectionSpecFactory.createIMSConnectionSpec();
     connSpec.setMetadataURL("class://opendb.DFSIVPDBView");
     // applications view of DB
     // created using the DLI Model Utility, generated from PSBGEN and DBDGEN source.
     connSpec.setDatastoreName(alias); // The IMS alias name defined in ODBM
     connSpec.setPortNumber(port); // The ICON DRDA port number
     connSpec.setDatastoreServer(host);
     // IP address or DNS name of the LPAR where ICON resides
     connSpec.setUser(user);
     connSpec.setPassword(pass);
     connSpec.setDriverType(4);
```

Next, we use the Connection Specification to allocate the PSB handler. See Example 6-15.

Example 6-15 Allocating PSB handle and access to DLI data

```
psb = PSBFactory.createPSB(connSpec);
// Get a handle of a PSB using the IMS Connection
// Specifications previously set
String pcbname = "PCB01";
PCB ivp1pcb = psb.getPCB(pcbname); // Get a PCB using the PSB you just created
// Allocate PSB1
psb.allocate(); // Allocate and establish a connection to the data
System.out.println("OpenDBIVP_main PSB " + pcbname + " for IVPDB1 Allocated");
// Do work on PSB1
SSAList ssaList = ivp1pcb.getSSAList("PhoneBook"); // SSA list to use in a DLI call.
// This SSA list qualifies the entire PhoneBook segment.
// PhoneBook is an alias name for segment A1111111 which was specified
// in the database view (DFSIVPDBView.java).
Path path = null; // Create a path object for later use
System.out.println("OpenDBIVP_main retrieving!!");
PathSet ps = ivp1pcb.batchRetrieve(ssaList);
// all the segment instances of PhoneBook.
// The data returned will be placed in
// a PathSet which is a collection of
// Path's containing the data you requested
System.out
      .println("OpenDBIVP_main Retrieved all segment instances of Phone Book");
System.out.println("FIRSTNAME\tLASTNAME\tEXTENSION\tZIPCODE");
System.out
      .println("------------------------------------------------------------------");
/*
 * The following while loop process the PathSet by checking that there is a next element
 * (ps.hasNext), then it prints out the three fields that are defined in the database
 * view (FIRSTNAME, LASTNAME, EXTENSION) for segment PhoneBook. This will continue until
 * there are no more elements in the PathSet
 */
while (ps.hasNext()) {
   path = ps.next();
   System.out.println(path.getString("FIRSTNAME").trim() + "\t\t"
         + path.getString("LASTNAME").trim() + "\t\t"
         + path.getString("EXTENSION").trim() + "\t"
         + path.getString("ZIPCODE").trim());
}
```

JCA 1.5 Common Client Interface Support

The JCA 1.5 Common Client Interface for IMS DB is new in IMS 11. With this support, we can use two methods to approach the DLI data, but as a first step in both cases, a connection has to be established with the data. The steps are:

1. Look up of ConnectionFactory instance using the Java Naming and Directory Interface (JNDI in a J2EE container), or build it from a ManagedConnectionFactory (non-J2EE).

   Note: In J2EEv5 (in WebSphere v7), we can also use "Dependency Injection" of the "ConnectionFactory" resource

2. Use the ConnectionFactory instance to get a connection to an IMS database (PSB).
3. Use the methods of SQLInteractionSpec, DLIInteractionSpec, or both.

This process is shown in Example 6-16 on page 201.

Example 6-16 Connection to IMS through a ConnectionFactory

```
  ========================================================================
    IMSManagedConnectionFactory mcf = new IMSManagedConnectionFactory();
    // Set connection properties
    mcf.setUser(user);
    mcf.setPassword(pass);
    mcf.setDatastoreName(alias);
    mcf.setDatastoreServer(host);
    mcf.setPortNumber(port);
    mcf.setMetadataURL("class://opendb.DFSIVPDBView");
    mcf.setSSLConnection(false);
    mcf.setDriverType(IMSManagedConnectionFactory.DRIVER_TYPE_4);
    mcf.setLoginTimeout(10);
    // Create CCI Connection
    ConnectionFactoryImpl cf = (ConnectionFactoryImpl) mcf
          .createConnectionFactory();
    connection = cf.getConnection();
    System.out.println("OpenDBIVPCCI_main connection created!!");
    /**/
    Interaction ix = connection.createInteraction();
```

At this point, we have the connection with the DLI database established and an *Interaction* object has been instantiated. This object has to be used with a *specification*, for which we have two possibilities:

- Using the SQLInteractionSpec class for issuing simple SQL SELECT calls against IMS DBs. See Example 6-17.

Example 6-17 Access with SQLInteractionSpec

```
    SQLInteractionSpec iSpec = new SQLInteractionSpec();
    iSpec.setSQL("SELECT FIRSTNAME, LASTNAME, EXTENSION, ZIPCODE"
          + " FROM PCB01.PhoneBook");
    System.out.println("OpenDBIVPCCISQL_main execute interaction!!");
    ResultSet results = (ResultSet) ix.execute(iSpec, null);
    System.out.println("FIRSTNAME\tLASTNAME\tEXTENSION\tZIPCODE");
    System.out
          .println("-------------------------------------------------------------------");
    while (results.next())
       System.out.println(results.getString("FIRSTNAME").trim()
             + "\t\t" + results.getString("LASTNAME").trim()
             + "\t\t" + results.getString("EXTENSION").trim() + "\t"
             + results.getString("ZIPCODE").trim());
    System.out
          .println("OpenDBIVPCCISQL_main Retrieved all segment instances of Phone Book");
    results.close();
    ix.close();
    connection.close();
```

- Using the DLIInteractionSpec class, for issuing simple DL/1 calls against IMS DBs, by specifying RETRIEVE, UPDATE, DELETE, or CREATE, and using traditional SSAs as a literal string. See Example 6-18 on page 202.

Example 6-18 Access with DLIInteraction

```
DLIInteractionSpec iSpec = new DLIInteractionSpec();
iSpec.setFunctionName("RETRIEVE");
iSpec.setPCBName("PCB01");
iSpec.setFetchSize(8);
// This query will return the FIRSTNAME, LASTNAME, EXTENSION, ZIPCODE
// for all PhoneBook entries
iSpec.setSSAList("PhoneBook");
// This SSA list qualifies the entire PhoneBook segment. PhoneBook is an
// alias name for segment A1111111 which was specificed in the database view
// (DFSIVPDBView.java).
// Create RecordFactory
RecordFactory rf = cf.getRecordFactory();
// Create Record
MappedRecord input = rf.createMappedRecord("PhoneBook");
// Specify the fields to retrieve
input.put("FIRSTNAME", null);
input.put("LASTNAME", null);
input.put("EXTENSION", null);
input.put("ZIPCODE", null);
System.out.println("OpenDBIVPCCIDLI_main execute interaction!!");
ResultSet results = (ResultSet) ix.execute(iSpec, input);
System.out.println("FIRSTNAME\tLASTNAME\tEXTENSION\tZIPCODE");
System.out
.println("-----------------------------------------------------------------");
while (results.next())
   System.out.println(results.getString("FIRSTNAME").trim()
         + "\t\t" + results.getString("LASTNAME").trim()
         + "\t\t" + results.getString("EXTENSION").trim() + "\t"
         + results.getString("ZIPCODE").trim());
System.out
      .println("OpenDBIVPCCIDLI_main Retrieved all segment instances of Phone Book");
results.close();
ix.close();
connection.close();
```

JDBC

IMS 11 supports JDBCv3. JDBC uses SQL-oriented queries, but based on the JDBC API. When using the JDBC programming interface with the IMS Universal DB resource adapter or the IMS Universal JDBC driver, your Java client application component has to do the following tasks:

- Look up a datasource instance using JNDI, in a managed environment (J2EE) or build it (non-J2EE).
- Use the datasource instance to obtain a Connection object for an IMS database (PSB).
- Use the JDBC API.

Figure 6-19 on page 203 shows what can done with the JDBC API. The ideal way is to work through a DataSource. The datasource contains the reference to the DLIDataBaseView class.

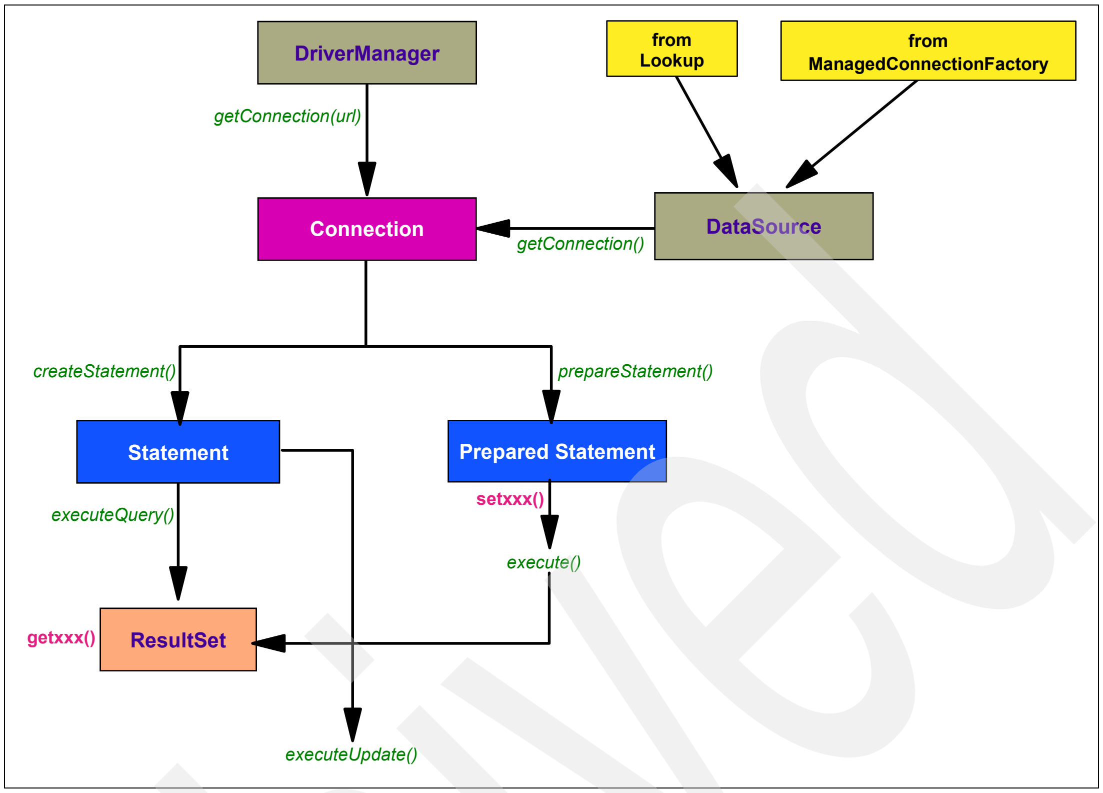


Figure 6-19   JDBC operations

Example 6-19. lists the steps in a non-J2EE program:

1. A DataSource object is instantiated.
2. The properties are set in this object with setters.
3. From the DataSource a connection is obtained.
4. A Statement is created on the connection.
5. An executeQuery is invoked on the statement with an SQLString as parameter, which returns a result set.
6. From the result set the result rows are printed.

Example 6-19   Retrieve using JDBC

```
......
//Setup DataSource (not in J2EE, we can NOT lookup, so we build the datasource)
      com.ibm.ims.jdbc.IMSDataSource ds = new com.ibm.ims.jdbc.IMSDataSource();
      ds.setMetadataURL("class://opendb.DFSSAM09DatabaseView");
      ds.setDatastoreName(alias);   //The IMS alias name defined in ODBM
      ds.setPortNumber(port);//The ICON DRDA port number
      ds.setDatastoreServer(host);//IP address or DNS name of the LPAR where
                                   //ICON resides
      ds.setUser(user);
      ds.setDriverType(4);
      ds.setPassword(password);
      try {
         //Establish a connection using the data source
         con = ds.getConnection();
         con.setAutoCommit(false);
         //Create a statement using the connection
```

```
        Statement st = con.createStatement();
        st.setFetchSize(10);
        ResultSet rs = st.executeQuery("SELECT PART,PARTDESC,AREA,DEPT, WORKORDER,ORDERQTY " +
              "FROM PARTSPCB1.PARTROOT, PARTSPCB1.STOKSTAT, PARTSPCB1.BACKORDR");
        System.out.println("Retrieve Parts that are on backorder");
        System.out.println("PART\t\tPART DESCRIPTION\tAREA\tDEPT\tWORKORDER\tORDERQTY");
        System.out.println("-----------------------------------------------------------");
        while(rs.next()){
           System.out.println(rs.getString("PART")+"\t"+
           rs.getString("PARTDESC")+"\t"+
           rs.getString("AREA")+"\t"+
           rs.getString("DEPT")+"\t"+
           rs.getString("WORKORDER")+"\t"+
           rs.getString("ORDERQTY")+"\t"
           );
        }
        rs.close();
        con.commit();

     } catch (SQLException e) {
     e.printStackTrace();
     }
  .....
```

6.5.2 ODBM access in with multiple IMS

A requirement is that the ODBM CSL address space has to be in the same LPAR as IMS or at least one member IMS of an IMSPLEX.

Figure 6-20 shows a possible configuration with one stand-alone IMS and one IMSPLEX with two members (with shared databases). The sharing IMSplex is composed of DBCTL2A, DBCTL2B. The stand-alone IMS is DBCTL1.

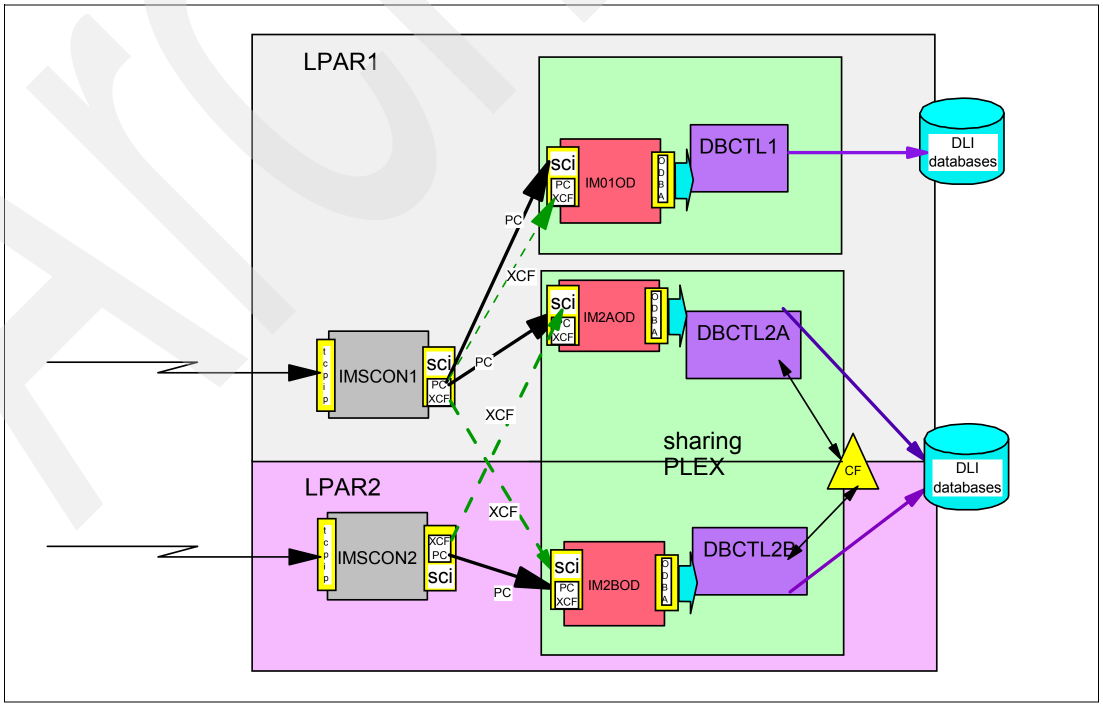


Figure 6-20 Example of a configuration

Two IMSCONs in two LPARs are used as gateway.

Between IMSCON and ODBM, communication is over SCI, which balances between: PCmode, if IMSCON is in the same LPAR; or XCF, if IMSCON is in a different LPAR.

The two IMSCONs and each IMS must belong to the same IMSPLEX (CSL group). When IMSCON starts up, connection is made to all members of the CSL group, and information is exchanged, so that IMSCON acquires knowledge of the real ODBM address spaces and the database component of the DBCTL and DB/DC systems they are in connection with, so that they can be addressed for DLI queries.

Important: All ODBM address spaces, even for unrelated IMSs, and the IMSCONs they communicate with must belong to the same CSL group.

6.6 IMS (DCCTL) as a service provider in SOA

IMS Data Communication (DCCTL), with IMS Connect as a frontend can be used as a Transaction server. New and existing transactions, running in the message processing regions of IMS can be triggered from remote and local clients on several platforms. This approach is possible from both J2EE and non-J2EE programs. The existing IMS programs can be part of Internet interactions, Web services, or can be called from whatever client program on a distributed or z/OS platform. Figure 6-21 gives an overview of the possibilities.

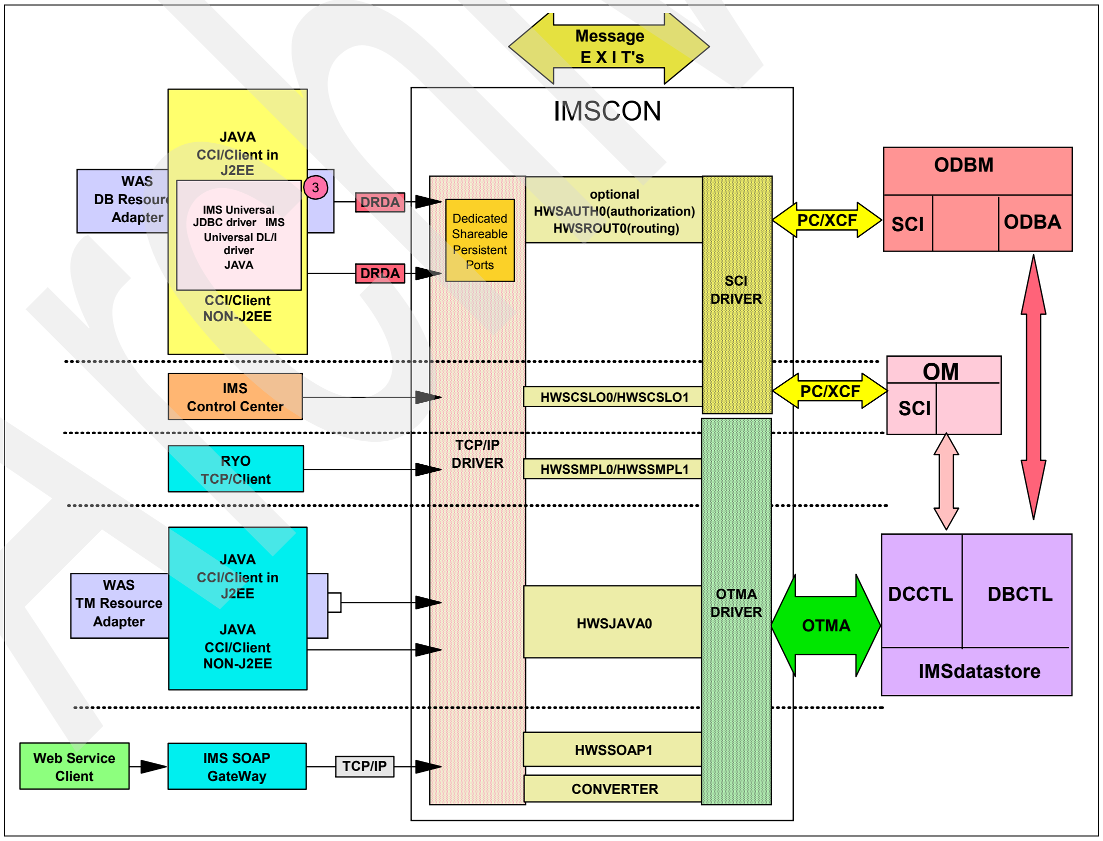


Figure 6-21 Call-in calls to IMS, IMS as Service Provider

All access is done through IMS Connect. Roughly the call-ins can be divided into three types:

- Access to IMS DLI databases through ODBM (SCI protocol)

  This type is new in IMS 11. It gives direct access to DLI databases.

- Calls to IMS Operation Management through OM (SCI protocol)

  IMS Control in a distributed DB2 Connect™ client can execute operator commands in an IMSPLEX.

- Access for IMS transactions (OTMA protocols). Requests from Web services over SOAP requires an additional front-end, the SOAP GateWay. See Figure 6-22 on page 207.

  More details can be found in *Powering SOA Solutions with IMS*, SG24-7662.

  We explain briefly several important items:

  – Access for IMS transactions through IMS connect can work in commit mode 0 (CM0) or CM1:

    • CM0 indicates that the commit of the transaction is done by IMS, and that the response is put on the output queue. This mode is also called commit-then-send.

    • CM1 means that the commit of the transaction is given by the client. This mode is also called send-then-commit.

  – Synclevel

    CM1 can work with three synchronization levels:

    • Level 0: Commit is issued as soon as the response is received by IMS Connect.

    • Level 1: A confirmation has to be sent back by the client.

    • Level 2:A *syncpoint* has to be issued by the client, eventually in combination with the usage of RR".

    CM0 always works without synclevels, but has two synchronizations options:

    • No synchronization: Used with IMS Connect

    • Synchronization: Used with WebSphere MQ

Note: IMS 11 is using the ACKTO timeout parameter for both CM1 and CM0 (included in IMS 11). If an ACK is not presented in time for CM0, the message is put in a special queue, default or as specified by the CM0ATOQ parameter.

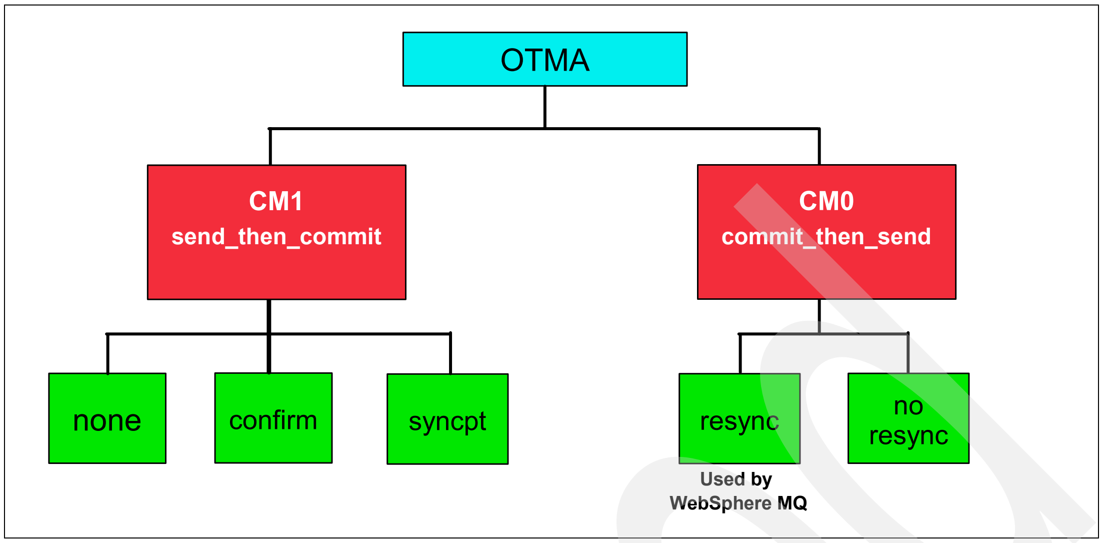


Figure 6-22 OTMA protocols

Call-in to an IMS as a Web Service provider

This access through the IMS SOAP Gateway has been improved in IMS 11. SOAP gateway improvements are discussed in 6.2, "IBM IMS Enterprise Suite SOAP Gateway" on page 180.

The IMS SOAP Gateway wizard, which becomes available with Rational Application Developer for System z version 7.6, offers the opportunity to prepare the gateway artifacts, in the same way as before, but with an additional selection for WS-enabling. Artifacts generated with WS-enabling, can only be used with Web Security.

In Example 6-20, we show the IMS Connect configuration, as it had to be augmented for transactional access and for SOAP. We enable SSL security, add the SOAP exit, and indicate the usage of an adapter.

Example 6-20 HWS config member for SOAP and security

```
HWS=(ID=IMOBHWS1,PSWDMC=N,RACF=N,RRS=Y,XIBAREA=50)
TCPIP=(EXIT=(HWSSMPL1,HWSSOAP1),HOSTNAME=TCPIP,
          PORTID=(7000,7001),SSLPORT=(7100),SSLENVAR=HWSSSL01)
DATASTORE=(GROUP=IMOBXCF,ID=IM1B,MEMBER=IM1BHWS1,TMEMBER=SCSIM1B,
          SMEMBER=IMOB)
DATASTORE=(GROUP=IMOBXCF,ID=IM2B,MEMBER=IM2BHWS1,TMEMBER=SCSIM2B,
          SMEMBER=IMOB)
ADAPTER=(XML=Y)
IMSPLEX=(MEMBER=IMOBHWS1,TMEMBER=PLEXB)
ODACCESS=(DRDAPORT=(ID=6000,KEEPAV=0,PORTTMOT=18000),
          DRDAPORT=(ID=6001,KEEPAV=0,PORTTMOT=18000),
          ODBMAUTOCONN=Y)
```

The SSLENVAR reference points to a file, also located in the PROCLIB. See Example 6-21

Example 6-21 file HWSSSL01

```
####################################################
#   SSL INTERFACE CONFIGURATION FILE FOR SOAPGW    #
####################################################
GSK_PROTOCOL_SSLV2=GSK_PROTOCOL_SSLV2_ON
GSK_PROTOCOL_SSLV3=GSK_PROTOCOL_SSLV3_ON
GSK_PROTOCOL_TLSV1=GSK_PROTOCOL_TLSV1_ON
GSK_KEYRING_FILE=IMS11
GSK_KEYRING_LABEL=IMSCONEV
GSK_KEYRING_PW=
GSK_KEYRING_STASH_FILE=
GSK_CLIENT_AUTH_TYPE=GSK_CLIENT_AUTH_FULL_TYPE
GSK_SESSION_TYPE=GSK_SERVER_SESSION
GSK_V2_CIPHER_SPECS=642
GSK_V3_CIPHER_SPECS=0906030201
DEBUG_SSL=ON
```

Other changes had to be made to member BPECFGHT, which is the configuration member for the BPE layer. At the end of the member, the lines shown in Example 6-22 were added.

Example 6-22 Configuration file for BPE with SOAP

```
......
......
# DEFINITIONS FOR SOAP GATEWAY
EXITMBR=(HWSEXIT0,HWS)
```

These lines point to another member, HWSEXIT0, as shown in Example 6-23.

Example 6-23 HWSEXIT0

```
EXITDEF(TYPE=XMLADAP,EXITS=(HWSXMLA0),ABLIM=8,COMP=HWS)
```

As a result of the redesign of the gateway, using AXIS2, the client code, which is using the new generated artifacts, has to be written in a different way.

The preparation of the gateway, with (or without) WS-security is done in two stages:

- With Rational Application Developer for System z Version 7.6

  A wizard in Rational Application Developer, takes as input the layouts of input/output messages. In the wizard we also specify, whether we want WS-security enabled or not.

  You also have to specify which port of the SOAP gateway will be addressed, HTTP or HTTPS. If WS-security is enabled, HTTPS is mandatory. The outcome of this wizard usage is several files, (WSDL interface file, correlator file) and a COBOL converter file, which has to send to the z/OS host. This file is used by IMS connect, called by the adapter for conversion of XML to non-XML and from non-XML to XML.

Note: When IMS is the service provider, the contents of request/response are determined by the layout of the input/output messages in IMS. Clients for the service must adapt to it. Messages between client and gateway are in SOAP format. The Web Service Description Language (WSDL) file, created by the Rational Application Developer for System z wizard, expresses in XML format, how this SOAP interface between client and gateway looks. The WSDL file can be taken as input to a wizard to produce a client proxy, usable by the client.

The IMS input/output messages in this case are mostly not in XML format, by this a converter is required in IMS connect, to be called by the adapter.

- With the deployment utility (on Windows or on z/OS)

  Correlator and WSDL interface must deployed under the SOAP gateway server. During this operation, the link is indicated between this correlator and the connection bundle entry to be used. Remember that the connection bundle entry contains the properties for the connection to IMS connect from the gateway.

  Note: All connection entries are in one connection bundle, but labeled. The correlator has to specify, which entry to use.

  The Rational Application Developer for System z wizard also produces the converter COBOL file. This has to be transferred to z/OS under all circumstances, compiled and made available as a loadable module for IMS connect. See Example 6-24.

Example 6-24 Compile, Linkedit of converter COBOL file

```
//SJSOAPCB  JOB CLASS=A,REGION=OM,MSGCLASS=X,MSGLEVEL=(1,1),
//   NOTIFY=&SYSUID
//*
//CLG   EXEC IGYWCL41
//COBOL.SYSIN  DD  DSN=VANAERS.SOURCE.COBOL(IMSPHBSD),DISP=SHR
//LKED.SYSLMOD DD  DSN=IMS11M.SDFSRESL,DISP=SHR
//*KED.SYSLMOD DD  DSN=VANAERS.IMS.LOAD,DISP=SHR
//LKED.SYSIN DD *
 ENTRY IMSPHBSD
 ALIAS IMSPHBSX
 NAME IMSPHBSD(R)
```

After the preparation activity we execute a Web Service stand-alone client, with and without WS-security. Remember that this currently requires two artifact generations and consequently will deploy.

1. Without Web Service security as shown in Example 6-25.

Example 6-25 Client code for IBM IMS Enterprise Suite SOAP Gateway (no security)

```
package imsphbk;
   /* files imported from generating, deploying the service */
import com.imsphbki.www.schemas.imsphbkiinterface.INPUTMSG;
import com.imsphbko.www.schemas.imsphbkointerface.OUTPUTMSG;
import com.imsphbki.www.schemas.imsphbkiinterface.INPUTMSGDocument;
import com.imsphbko.www.schemas.imsphbkointerface.OUTPUTMSGDocument;
import files.target.*;

public class TestClient {
    /**
     * @param args
     */
    public static void main(String[] args) {
       INPUTMSGDocument inputdoc = INPUTMSGDocument.Factory.newInstance();
       INPUTMSG input = inputdoc.addNewINPUTMSG();
       input.setInLl((short) 32);
       input.setInZz((short) 0);
       input.setInTrcd("IVTNO ");
       input.setInCmd("DISPLAY");
       input.setInName1("LAST1");
       input.setInName2("");
       input.setInZip("");
       input.setInExtn("");
       inputdoc.setINPUTMSG(input);
       // Invoke PhoneBook Service
       try {
          String target = "http://localhost:8080/imssoap/services/IMSPHBKService";
          IMSPHBKServiceStub phonebookservice = new IMSPHBKServiceStub(target);
          OUTPUTMSGDocument outputdoc = phonebookservice
                .iMSPHBKOperation(inputdoc);
          OUTPUTMSG output = outputdoc.getOUTPUTMSG();
          // Populate output values to GUI
          System.out.println("LL: " + output.getOutLl());
          System.out.println("ZZ: " + output.getOutZz());
          System.out.println("Message: " + output.getOutMsg());
          System.out.println("Last name: " + output.getOutName1());
          System.out.println("First name: " + output.getOutName2());
          System.out.println("Extension: " + output.getOutExtn());
          System.out.println("Zip: " + output.getOutZip());
       } catch (Exception iconex) {
          System.err.println("Exception: " + iconex.toString());
       }
    }
}
```

2. With Web Service security

   This execution has the following requirements:

   – The gateway artifacts must be regenerated and deployed, with WS enabled.
   – It is required to use the HTTPS protocol between the client application and the SOAP Gateway. As a minimum, this requires a setup of a *truststore* on the client side and a *keystore* on the gateway side.
   – Usage of SSL between SOAP Gateway and IMS Connect is not mandatory. This option is indicated by the *correlation* bundle selection of the adequate entry in the *connection bundle*. See Figure 6-23 on page 211.

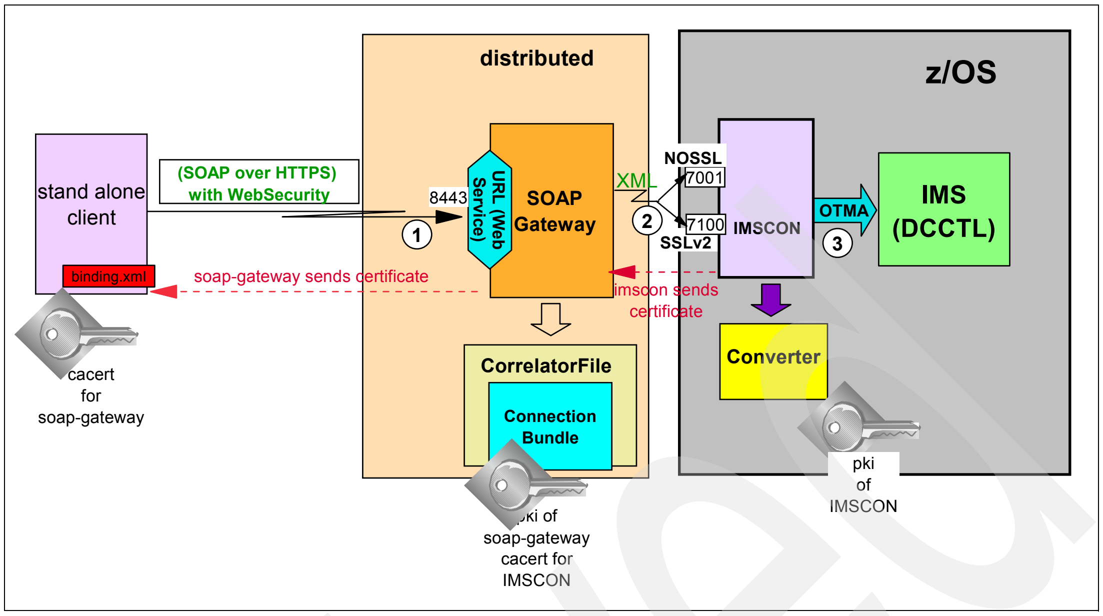


Figure 6-23 Web service client with WS-security

– All connection entries are in one *connbundle.xml* file. See the setup of one entry in Example 6-26.

Example 6-26 Connection bundle entry POK42S

```
<root>
....
   <conn>

      <connBundleName>POK42S</connBundleName>
      <connHostName>wtsc42.itso.ibm.com</connHostName>
      <connPortNum>7100</connPortNum>
      <connDataStoreName>IM1B</connDataStoreName>
      <connUserID></connUserID>
      <connPassword></connPassword>
      <connGroupName></connGroupName>
      <sslKeystoreName>C:/Program Files/IBM/IMS Enterprise Suite V1/SOAP 
Gateway/rsa/server.keystore.ks</sslKeystoreName>
      <sslKeystorePasswd>imssoap</sslKeystorePasswd>
      <sslTruststoreName>C:/Program Files/IBM/IMS Enterprise Suite V1/SOAP 
Gateway/rsa/server.keystore.ks</sslTruststoreName>
      <sslTruststorePasswd>imssoap</sslTruststorePasswd>
      <sslEncrypType>strong</sslEncrypType>
      <call-outTPipes></call-outTPipes>
      <call-outSslKeystoreName></call-outSslKeystoreName>
      <call-outSslKeystorePasswd></call-outSslKeystorePasswd>
      <call-outSslTruststoreName></call-outSslTruststoreName>
      <call-outSslTruststorePasswd></call-outSslTruststorePasswd>
      <call-outBasicAuthName></call-outBasicAuthName>
      <call-outBasicAuthPasswd></call-outBasicAuthPasswd>
   </conn>
</root>
```

– The currently supported WS-security is UserNameToken-based. The SOAP gateway client `binding.xml` file, has to indicate by handler, how the user ID and password have to be picked up. Figure 6-24 shows where this binding file is located in the IMS SOAP Gateway directory.

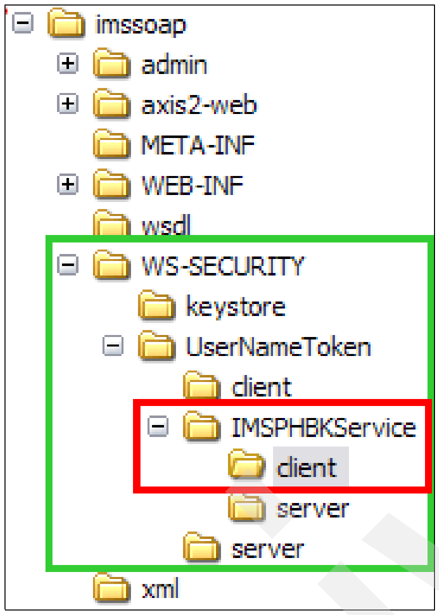


Figure 6-24 Location of client binding file for WS-security

In Example 6-27, notice the `tokenGenerator`, which indicates the handler for collecting the input. Other handlers can be specified here.

Example 6-27 binding.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<securityBindings
xmlns="http://www.ibm.com/xmlns/prod/websphere/200608/ws-securitybinding">
    <securityBinding name="application">
        <securityOutboundBindingConfig>
           <tokenGenerator name="gen_unametoken"
classname="com.ibm.ws.wssecurity.wssapi.token.impl.CommonTokenGenerator">
             <valueType
localName="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1
.0#UsernameToken" uri="" name="Username Token"/>
             <securityTokenReference reference="request:uname_token"/>
             <callbackHandler
classname="com.ibm.websphere.wssecurity.callbackhandler.UNTGenerateCallbackHandler">
                <basicAuth userid="VANAERS" password="xxxxxxxxx"/>
             </callbackHandler>
             <jAASConfig configName="system.wss.generate.unt"/>
           </tokenGenerator>
        </securityOutboundBindingConfig>
    </securityBinding>
</securityBindings>
```

– The client program has to be prepared for HTTPS. This requires changes and additions to the program in Example 6-25 on page 209. The excerpt of the changes is shown in Example 6-28 on page 213.

Example 6-28 Excerpt of changes

```
........
public class TestClientSecTran {
   /**
    * @param args
    */
   public static void main(String[] args) {
      try {
         System.setProperty("java.net.ssl.trustStore",
      //    "C:\\Program Files\\IBM\\IMS Enterprise Suite V1\\SOAP
Gateway\\java\\jre\\lib\\security\\cacerts");
      "C:/Program Files/IBM/IMS Enterprise Suite V1/SOAP
Gateway/java/jre/lib/security/cacerts");

         //System.setProperty("java.net.ssl.trustStoreType","JKS");
         System.setProperty("java.net.ssl.trustStorePassword","changeit");
         Security.addProvider(new com.ibm.jsse.IBMJSSEProvider());
      } catch (Exception ex1) {
         System.err.println("Exception1: " + ex1.toString());
      }
      INPUTMSGDocument inputdoc = INPUTMSGDocument.Factory.newInstance();
      INPUTMSG input = inputdoc.addNewINPUTMSG();
      input.setInLl((short) 32);
      input.setInZz((short) 0);
      input.setInTrcd("IVTNO ");
      input.setInCmd("DISPLAY");
      input.setInName1("LAST1");
      input.setInName2("");
      input.setInZip("");
      input.setInExtn("");
      inputdoc.setINPUTMSG(input);
      // Invoke PhoneBook Service
      try {
         String target = "https://localhost:8443/imssoap/services/IMSPHBKService";
.........
```

Example 6-29 shows the SOAP envelope, with header and body, that is sent to gateway.

Example 6-29 Input SOAP envelope

```
<?xml version="1.0" encoding="utf-8" ?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Header xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <s:Security
xmlns:s="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
soapenv:mustUnderstand="1">
         <s:UsernameToken u:Id="unt_20">
            <s:Username>VANAERS</s:Username>
            <s:Password
Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#Passwor
dText">XXXXXXXX</s:Password>
         </s:UsernameToken>
      </s:Security>
      <wsa:To>https://localhost:8443/imssoap/services/IMSPHBKService</wsa:To>
      <wsa:MessageID>urn:uuid:C3F7D4DEBF4EBA2E831250525230603</wsa:MessageID>
      <wsa:Action>urn:IMSPHBK</wsa:Action>
   </soapenv:Header>
   <soapenv:Body>
      <ims:INPUTMSG xmlns:ims="http://www.IMSPHBKI.com/schemas/IMSPHBKIInterface">
         <ims:in_ll>32</ims:in_ll>
         <ims:in_zz>0</ims:in_zz>
         <ims:in_trcd>IVTNO</ims:in_trcd>
         <ims:in_cmd>DISPLAY</ims:in_cmd>
         <ims:in_name1>LAST1</ims:in_name1>
         <ims:in_name2></ims:in_name2>
         <ims:in_extn></ims:in_extn>
         <ims:in_zip></ims:in_zip>
      </ims:INPUTMSG>
   </soapenv:Body>
</soapenv:Envelope>
```

The typical SOAP items, like the header with security and name space definitions, are handled by the gateway.

The data part of the envelope continues with IMS Connect, where the data, which is still in XML format, will be converted to the IMS message.

6.7 IMS (DCCTL) as a service consumer in SOA

OTMA, together with hold-queue-capable OTMA clients, such as IMS Connect, supports call-out requests from IMS application programs running in IMS dependent regions to data or service providers that are external to the IMS installation.

When an IMS application program makes a call-out request, IMS can be viewed as a client in a client-server relationship where the server is the external application to which IMS is making the call-out request. An overview of the possibilities is shown in Figure 6-25.

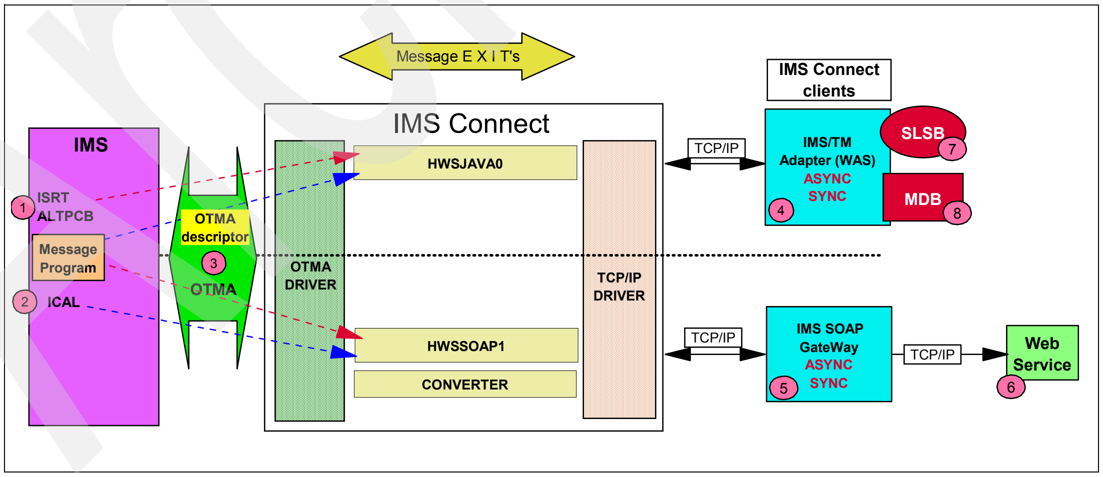


Figure 6-25 IMS call-outs

We distinguish the following possibilities, which can be synchronous and asynchronous:

- Call to a stateless session bean (SLSB) in WebSphere Application Server. An SLSB is an enterprise bean (EJB component) that provides a stateless service to the client.
- Call to a message-driven bean (MDB) in WebSphere Application Server. An MDB is an enterprise bean that allows J2EE applications to process messages asynchronously
- Web Services (SOAP) call through the SOAP Gateway

The difference between a synchronous and an asynchronous call is:

- Synchronous

  The program (Message Processing Program (MPP), Batch Messaging Program (BMP), Java Batch Program (JBP), Java Message Program (JMP)) issuing the outbound request in IMS, waits for the response.
- Asynchronous

  The issuing program (MPP, BMP, JBP, JMP) does not wait; the response is returned as a call-in, and it becomes a new transaction which can be picked up by a new transaction driven program(MPP, BMP, JBP, JMP).

The Synchronous and Asynchronous call-out methods do not preclude that you can send out messages in other ways. Among those possibilities, we mention sending and receiving messages by WebSphere MQ Series, or through DB2 stored procedures.

The difference between a synchronous and an asynchronous call is:

- Synchronous: The issuing program (MPP, BMP, JBP, JMP) in IMS waits for the response
- Asynchronous: The response is returned as a call-in; it becomes a new transaction.

IMS application programs running in IMS-dependent regions can call out through the Open Transaction Manager Access (OTMA) component of IMS to servers that are outside the IMS installation to request data or services. For both types of call-out request, IMS Connect serves as the TCP/IP gateway (server) between the IMS Connect client that use TCP/IP and the OTMA component of IMS.

An IMS configuration that supports such call-out requests requires multiple components. Exactly which components are required in addition to OTMA differs slightly depending on whether the call-out requests are processed synchronously or asynchronously.

Looking at Figure 6-25 on page 214 you should realize that IMS Connect always acts as a server and that the Enterprise JavaBeans in WebSphere Application Server and the SOAP Gateway act as a client, retrieving the call-out message from a *hold* tpipe. OTMA destination descriptors must be defined to route the call-out request to the tpipe of the IMS Connect client that is configured to retrieve the call-out requests.

Those technologies are described in *Powering SOA Solutions with IMS*, SG24-7662.

Certain technologies were improved in IMS 10 (SPE) or in IMS 11. In this book, we highlight the synchronous call-out, which is the most recent technology to join the SOA call-outs.

6.7.1 Synchronous call-out

Synchronous call-out requests are processed in real time and travel from the IMS application program running in an IMS dependent region, out to an external data or service provider, and back to the IMS application program while the IMS application program remains scheduled in the IMS dependent region.

For synchronous call-out requests, an IMS application program running in an IMS dependent region issues the DL/I ICAL call and waits in the dependent region to process the response also. When the ICAL call is issued, IMS generates a correlation token for synchronous call-out requests and routes the request to an OTMA destination. The correlation token is included with the call-out request and must be returned to IMS with the response to route the response back to the requesting IMS application program.

Artifacts in WebSphere Application Server, SLSB and MDB are able to act as the IMS Connect client for synchronous call-out (service provider for IMS as service consumer). Web Services also can act as the service provider through the SOAP Gateway.

Call-out request routing

OTMA routes the call-out request to the appropriate client tpipe queue based on destinations defined by using one of the following methods:

- OTMA destination descriptor

  The OTMA ALTPCB destination descriptor specifies a destination for OTMA ALTPCB output sent to either IMS Connect or non-OTMA destinations such as an SNA terminal or printer. The OTMA ALTPCB destination descriptor also supports the routing of call-out requests to services connected to by the IMS TM Resource Adapter and IMS SOAP Gateway.

  OTMA ALTPCB destination descriptors are coded in the DFSYDTx PROCLIB member and are identified by a D in column one of the descriptor entry.

  The ALTPCB destination attributes that you can specify include:

  – The destination type: The OTMA ALTPCB destination descriptor supports only IMS Connect, tagged by IMSCON, and non-OTMA destinations, tagged by NONOTMA, as destination types.
  – For IMS Connect destinations: An OTMA tmember name or an OTMA super member name and an optional tpipe name.
  – For IMS Connect XML conversion support for IMS SOAP Gateway, an XML adapter name and an XML converter name.

  When specifying a destination name in an OTMA ALTPCB descriptor (see Example 6-30), you can generalize destination names by using an asterisk (*) as a wildcard character or as a mask of the final characters in the destination name field.

  Example 6-30 Sample DFSYDTx with OTMA descriptors

```
Sample OTMA Descriptor for call-out to IMS TM Resource Adapter
D WASDEST1 TYPE=IMSCON TMEMBER=HWS1 TPIPE=WASTP1 SYNTIMER=5000
Sample OTMA Descriptor for call-out to IMS SOAP Gateway
D SGDEST1 TYPE=IMSCON TMEMBER=HWS1 TPIPE=SGTP1 SYNTIMER=5000
D SGDEST1 ADAPTER=HWSXMLA0 CONVERTR=XMLCNV1D
```

- Type-2 commands

  You can dynamically add, update, or delete OTMA destination descriptors using the following type-2 commands:

  – CREATE OTMADESC
  – UPDATE OTMADESC
  – DELETE OTMADESC

  A restart of IMS is not required; in addition, any changes that you make by using these type-2 commands persist across warm and emergency restarts.

Synchronous call-out request as (CM0) output messages

For synchronous call-out requests from IMS application programs that issue the DL/I ICAL call, OTMA processes the request as a CM0 output message.

Note: OTMA sets the CM0 flag in the state data prefix and waits for an acknowledgement of receipt of the synchronous call-out message. If an acknowledgement is not returned within the ACK timeout interval, OTMA takes the following action:

1. Issues return code X'100' and reason code X'104' to the IMS application program.
2. Discards the synchronous call-out request message. The message is not rerouted.
3. Issues DFS3494E to the z/OS system console.

You can specify a timeout interval for an OTMA client in the OTMA client descriptor. You can override the timeout interval in an OTMA client descriptor by specifying a different interval in the TIMEOUT parameter of the /START TMEMBER command. You can also override the timeout interval in the OTMA client descriptor by specifying a smaller timeout interval in the OTMA client's client bid request.

For individual transactions, you can only specify a timeout interval that is shorter than the timeout interval specified for the OTMA client.

If you do not specify a send-then-commit timeout value, OTMA uses a default value of 120 seconds. To view the current timeout interval set for an OTMA client, issue the /DISPLAY TMEMBER command.

An ACK message frees the tpipe queue to send and receive other call-out messages and the IMS application program continues to wait in the dependent region until either a response to the synchronous call-out message is received or the synchronous call-out request times out.

A NAK message frees the tpipe queue, and also tells OTMA what to do with the synchronous call-out request message and whether to maintain or end the current RESUME TPIPE call:

- Discard the rejected synchronous call-out request message and terminate the RESUME TPIPE connection.
- Discard the rejected synchronous call-out request message, but maintain the RESUME TPIPE connection to continue retrieving other synchronous call-out request messages.
- Keep the rejected synchronous call-out request message on the tpipe queue, but terminate the RESUME TPIPE connection.

Stateless session bean (SLSB) and message-driven bean (MDB) artifacts in WebSphere Application Server and the SOAP Gateway are able to act as the IMS Connect client for synchronous call-out.

Enhancements were required in the IMS TM Resource Adapter for enabling an IMS application to synchronously call out to the J2EE artifacts running inside the WebSphere Application Server. IMS TM Resource Adapter is being enhanced with:

- JCA 1.5 Inbound API
- Synchronous inbound communication with IMS
- Concurrent call-out message processing
- SLSB and MDB programming models

To receive and respond to synchronous call-out request messages from an IMS application, IMS TM Resource Adapter uses the enhanced Resume TPIPE and Send-Only function. IMS TM Resource Adapter issues a Resume TPIPE request to a tpipe as specified by the application.

DL/I call ICAL in the IMS program

An IMS application issues the DL/I ICAL SENDRECV call to send the call-out request data and an OTMA descriptor name. The specified OTMA descriptor name matches one of the descriptors that is configured with IMS and it consists of routing information such as the TMEMBER and TPIPE name.The new ICAL call has the following syntax:

```
ICAL--aib--request_area--response_area
```

AIB parameter

The AIB parameter specifies the application interface block (AIB) to be used for this call. This parameter is an input and output parameter. It contains the following fields:

- AIBID

  Eyecatcher. This 8-byte field must contain DFSAIBbb.

- AIBLEN

  AIB length. This field must contain the actual length of the AIB that the application program obtained.

- AIBSFUNC

  Sub function code. This field must contain SENDRECV. The IMS application program uses this sub function for synchronous program to program communication. The IMS application program can send a message and wait for the response.

- AIBRSNM1

  Resource name. This field contains the OTMA descriptor name. AIBRSNM1 is an 8-byte alphanumeric left-justified field that is padded with blanks.

- AIBRSFLD

  This 4- byte field is used by the IMS application program to specify a time value to wait in one-hundredths of a second for the synchronous call process to complete. This value overrides the value specified in the OTMA descriptor. The minimum value is 0 and the maximum value is 999999. The maximum value will be used for any value larger than 999999. The system default is 10 seconds.

- AIBOALEN

  Request area length. This is an input and an output parameter. As an input parameter, this 4 -byte field must contain the length of the input request area that is specified in the call list. As an output parameter this field is only updated when partial data is returned (AIB return code x'100', AIB reason code x'00C'). In the case of partial data returned, this field contains the actual length of the response message. For any return code other than partial data, this field is unchanged.

- AIBOAUSE

  Response area length. This is an input and an output parameter. As an input parameter this 4-byte field contains the length of the output response area that is specified in the call list. As an output parameter, this field is updated by IMS with the length of the response message returned in the response area. When partial data is returned because the response area is not large enough, AIBOAUSE contains the length of data returned in the response area and AIBOALEN contains the actual length of the response message.

- AIBRETRN

  Indicates the AIB return code.

- AIBREASN

  Indicates the AIB reason code.

The message is then routed and available in the TPIPE.

OTMA delivers the request message through IMS Connect. A correlation token will be sent together with the call-out request.

After the send is successful, OTMA issues the IWAIT to start the timer to wait for the response message. The IMS application synchronously waits for the response.

Finally, IMS correlates the message back to the corresponding IMS transaction instance.

6.7.2 Synchronous call-out service providers

Call-out requests from IMS can address the following service providers:

- Message-driven bean (MDB) in WebSphere Application Server
- Stateless session bean (SLSB) in WebSphere Application Server
- Web Services, where they are available, through the SOAP Gateway

OTMA provides the following functions to the IMS Synchronous call-out SPE:

- Flow the synchronous call-out request and receive the response
- Enhance the OTMA output descriptor
- Provide the timeout function for the ICAL call
- Provide the OTMA protocol updates so that the synchronous call-out request and the response can be recognized by both IMS Connect and OTMA
- Provide a simplified OTMA TPIPE log record to track input and output flow of a synchronous call-out message
- Provide the OTMA command updates

OTMA connects the ICAL call to the IMS Connect client applications so that a synchronous call-out request can be delivered to them and the response can be given back to the IMS application.

This synchronous call-out request is delivered through the OTMA Resume TPIPE method. If the request is larger than 32 KB, it is delivered to IMS Connect by using the multi-segment method. The size of each message segment is equal to or less than 32 KB. IMS Connect assembles the segments and delivers them as one complete message to IMS Connect client applications, such as IMS TM Resource Adapter and IMS SOAP Gateway.

A response message for a synchronous call-out request must come back to OTMA as a special form of send-only message. This type of send-only message has no trancode and can be a regular response data or error information to be passed back to the ICAL call. The response message can also be a multi-segment message. OTMA assembles the multi-segment messages into a single message to be returned back to the IMS application.

OTMA descriptor includes a new parameter for the timeout value of a synchronous call-out request. This optional parameter can be used if no timeout value is specified on the ICAL call. However, if no timeout value is specified on both OTMA descriptor and ICAL call, OTMA uses a system default value of 10 seconds to timeout the call.

The DL/I test program (DFSDDLT0) is enhanced to support the ICAL DL/I call, so you can use DFSDDLT0 to test the synchronous call-out request.

Call-out to MDB in WebSphere Application Server as service provider

A message-driven bean (MDB), MDBv2, written in Java and deployed in WebSphere Application Server, allows its application code to synchronously receive messages delivered to a JMS destination. An MDBv2 is an enterprise Java bean which can be activated by message delivery and then consume and process messages.

The extended MDBv2.1 can be called directly (no JMS involved) if appropriate adapters are installed. The *V10SPE Transaction Manager Resource Adapter (TMRA)* offers this facility. The MDB has the advantage of being simple. Figure 6-26 presents the interaction with a MDB. For synchronous call the MDB Java code is triggered in the onMessage() method.

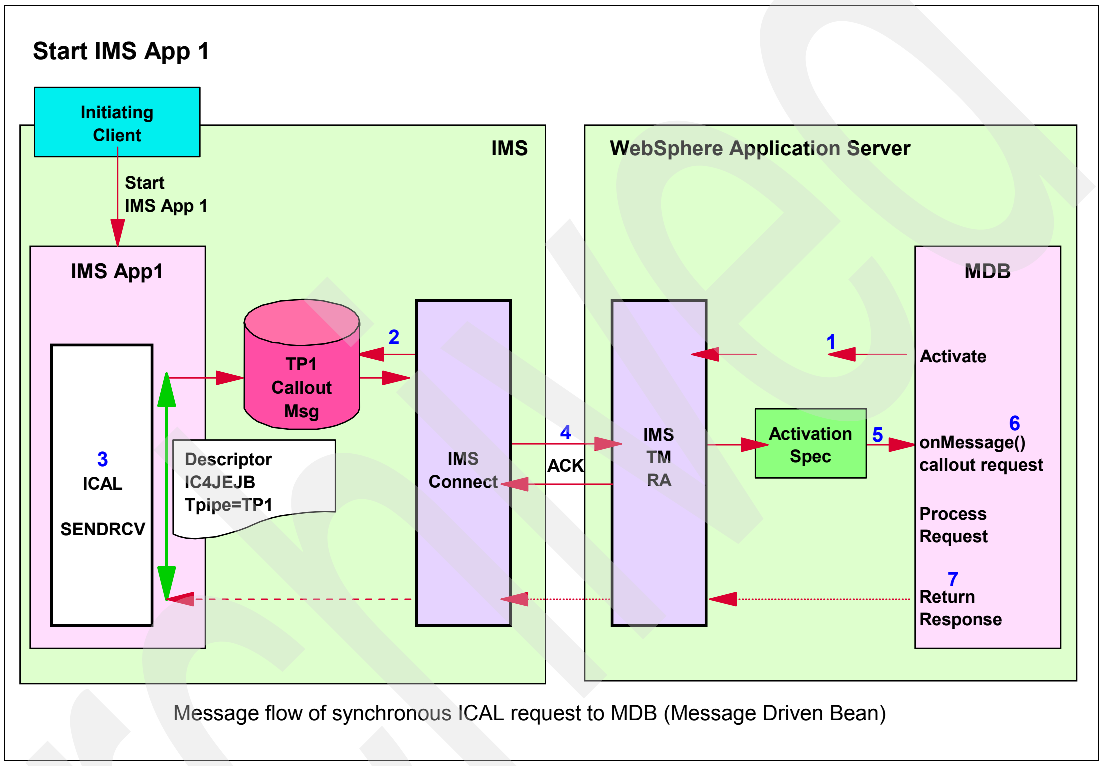


Figure 6-26 Synchronous call-out to MDB using IMS TM Resource Adapter

During deployment and configuration of an MDB in WebSphere Application Server, the MDB deployer has to specify the tpipe name for the call-out requests through the *ActivationSpec*, which is a resource description in WebSphere Application Server, established with the WebSphere Application Server administration facilities. The sequence is as follows:

1. When the J2EE application containing the deployed MDB is started in the WebSphere Application Server through the *ActivationSpec* and the underlaying listening software, a sharable persistent connection to IMS Connect is obtained through TMRA. It issues a `SYNC_RECEIVE_ASYNCOUTPUT_SINGLE_WAIT` interaction, specifying the tpipe name as the value for the alternate client ID, and sets a very large timeout value. TMRA issues a RESUME TPIPE request to the tpipe and waits for the call-out request from IMS Connect. The addressing information is pulled from properties in the MDB deployment descriptor.
2. If a call-out request is not available at the time of the `SYNC_RECEIVE_ASYNCOUTPUT_SINGLE_WAIT` request, TMRA will be blocked.
3. An initiating client, such as a terminal or an IMS Connect or OTMA client provokes the scheduling of an IMS application. The IMS application issues an ICAL call to an OTMA destination routing descriptor, which contains the destination tpipe name. The call-out request message is queued in this tpipe. When the DL/I ICAL call is issued, IMS

generates a correlation token for synchronous call-out requests. This correlation token is included with the call-out request and must be returned to IMS with the response to route the response back to the requesting IMS application program. See Example 6-31.

Example 6-31 OTMA Descriptor for call-out to IMS Resource Adapter

```
D WASDEST1 TYPE=IMSCON TMEMBER=HWS1 TPIPE=WASTP1 SYNTIMER=5000
```

4. When the call-out request is available in the tpipe, IMS Connect delivers the call-out message to TMRA.
5. TMRA receives the call-out request message and passes the call-out request to the MDB. The MDB receives the message in the onMessage() method and processes the call-out request. Response is returned over IMS Connect to the waiting program.

Note: An MDB directed call-out requires the installation of the Transaction Management Resource Adapter (TMRA) in the WebSphere Application Server.

Each MDB is always paired with a “ActivationSpec”, to specify the conditions in which its instance gets activated. For traditional MDBs v2, this was through the arrival of a message on some queue. MDB v2.1 extends the activation concept through new adapters, an MDB can now be called.

Input data are received and response data are returned as an object of type *javax.resource.cci.Record.*

For a code example look in *Powering SOA Solutions with IMS,* SG24-7662.

Call-out to SLSB in WebSphere Application Server as service provider

A stateless session bean (SLSB), written in Java and deployed in WebSphere Application Server, allows its application code to be called from a remote client. IMS Connect V10SPE can be this remote client with the assistance of the new V10SPE TMRA, shown in Figure 6-27.

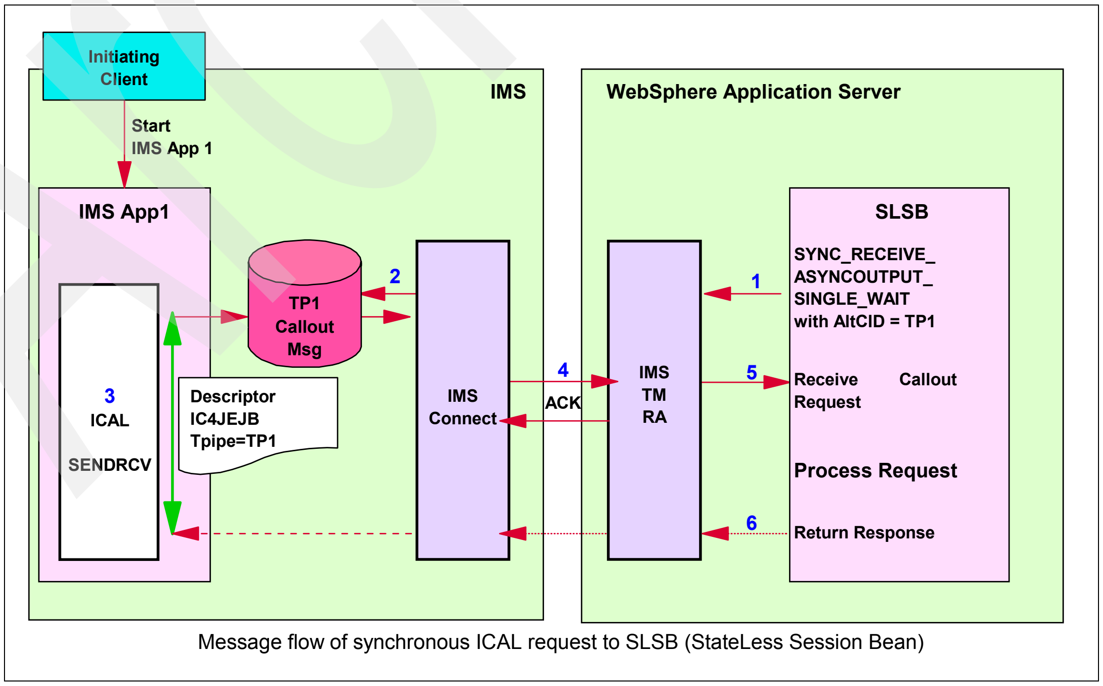

Initiating Client
IMS
WebSphere Application Server
Start IMS App 1
IMS App1
SLSB
SYNC_RECEIVE_ ASYNCOUTPUT_ SINGLE_WAIT with AltCID = TP1
TP1 Callout Msg
2
1
3
ICAL
SENDRCV
Descriptor IC4JEJB Tpipe=TP1
IMS Connect
4
ACK
IMS TM RA
5
Receive Request
Callout
Process Request
6
Return Response
Message flow of synchronous ICAL request to SLSB (StateLess Session Bean)

Figure 6-27 Message flow of synchronous call-out to SLSB using TMRA

In the figure:

1. The EJB application in WebSphere Application Server starts and obtains a sharable persistent connection to IMS Connect through TMRA. It issues a `SYNC_RECEIVE_ASYNCOUTPUT_SINGLE_WAIT` interaction, specifying the tpipe name as the value for the alternate client ID, and sets a very large timeout value. TMRA in turn issues a RESUME TPIPE request to the tpipe and waits for the call-out request from IMS Connect.

   Attention: SLSB is not triggered in the same way as MDB. The SLSB has to be started so that it can contact IMS Connect and retrieve the call-out messages. This can done with two types of startup beans:

   - A *module startup bean* is a session bean that is loaded when an EJB JAR file starts. Module startup beans enable Java Platform Enterprise Edition (Java EE) applications to run business logic automatically, when an EJB module starts or stops normally.
   - An *application startup bean* is a session bean that is loaded when an application starts. Application startup beans enable Java EE applications to run business logic automatically, when an application starts or stops normally.

   For a code example, refer to *Powering SOA Solutions with IMS,* SG24-7662.

2. If a call-out request is not available at the time of the `SYNC_RECEIVE_ASYNCOUTPUT_SINGLE_WAIT` request, TMRA will be blocked and the EJB waits for the next available call-out message or until timeout occurs.
3. The IMS application issues an ICAL call to an OTMA destination routing descriptor, which contains the destination tpipe name. See Example 6-32. The call-out request message is queued in this tpipe.

   Example 6-32 Sample OTMA Descriptor for call-out to IMS TM Resource Adapter

   ```
   D WASDEST1 TYPE=IMSCON TMEMBER=HWS1 TPIPE=WASTP1 SYNTIMER=5000
   ```

4. When the call-out request is available in the tpipe, IMS Connect delivers the call-out message to the IMS TM Resource Adapter.
5. TMRA receives the call-out request message and forwards the call-out request to the EJB. The EJB processes the call-out request.
6. A response message for a synchronous call-out request return to OTMA as a special form of SYNC_SEND message. This type of send-only message has no trancode and can be a regular response data or error information to be passed back to the ICAL call. The response message can also be a multi-segment message. OTMA assembles the multi-segment messages into a single message to be returned back to the IMS application.

Note: Compare the MDB and the SLSB model:

- The SLSB application has the responsibility to manage the call-out protocol (such as the Resume TPIPE and Send-Only) with IMS Connect and IMS to retrieve the call-out requests. With MDB, IMS TM Resource Adapter manages the call-out protocol and delivers the requests to the MDB.
- EJB has to manage the correlation token and send it back with the response message. For MDB, IMS TM Resource Adapter manages the correlation token internally and is transparent to the MDB application.
- MDB model requires J2EE Application Server that supports JCA 1.5 with inbound communication.

Call-out to a Web Service as service provider

A WebSphere Application Server can be the Web Service provider, but the service is also delivered by many other external providers. Participating in the Web Service arena, means also that IMS (IMS SOAP Gateway) have to keep up with the latest standards and quality of service (QoS). IBM IMS Enterprise Suite SOAP Gateway, used with this solution, starts to implement QoS that is based on a policy set; it can implement user name and password ws-security. Figure 6-28 shows the Web Services interaction.

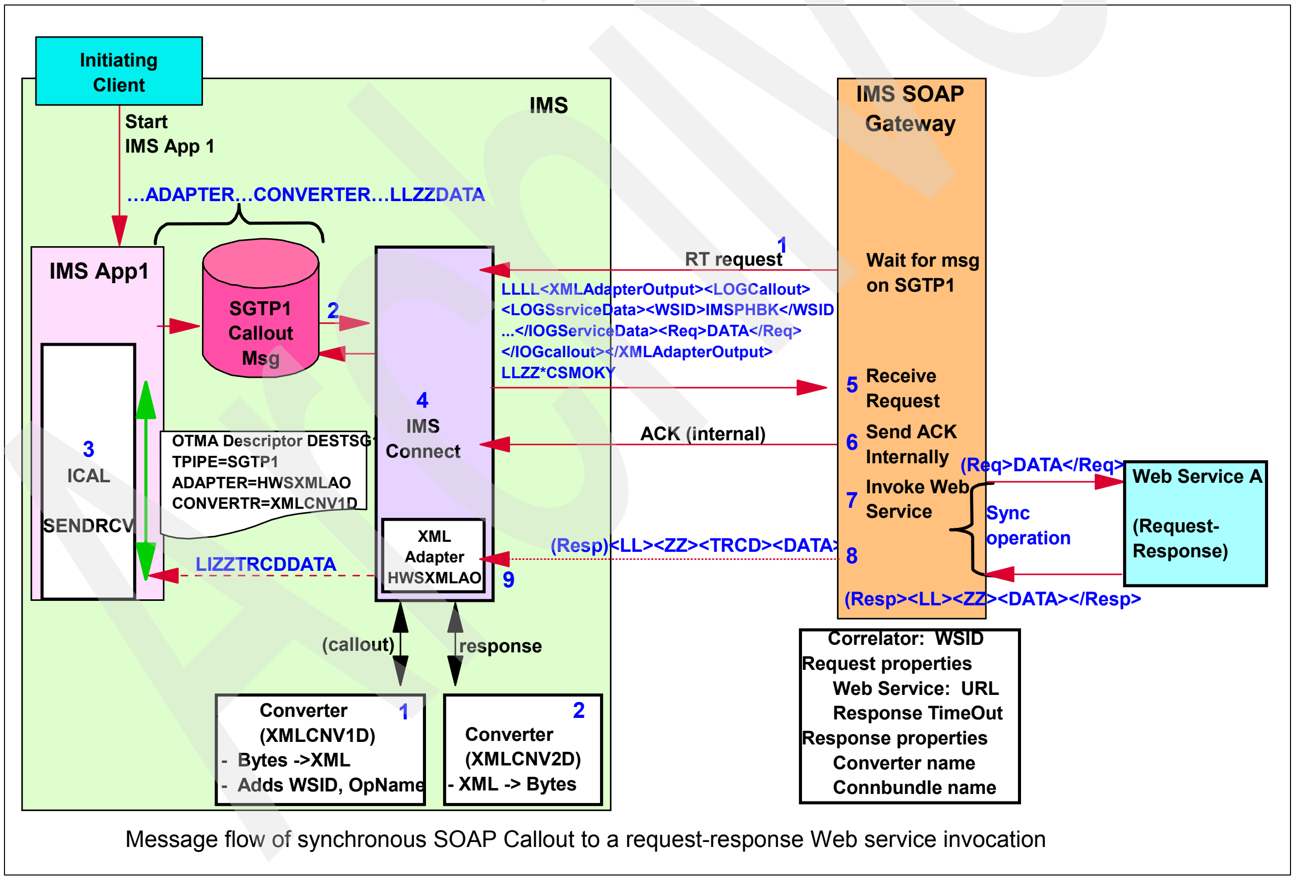


Figure 6-28 Accessing Web Services synchronously from IMS

Figure 6-28 on page 223 shows the following steps of the interaction:

1. A new sharable connection is first established with IMS Connect, and then IMS SOAP Gateway sends a Resume TPIPE request and waits for IMS Connect to send back the call-out message.
2. If no call-out message is available at the time of the Resume TPIPE request, IMS SOAP Gateway sleeps for a default number of seconds before issuing another Resume TPIPE request to IMS Connect to check for the call-out request message. This sleep time (also called the poll interval time) is configurable by the user with the IMS SOAP Gateway deployment utility.
3. The IMS application issues an ICAL call to an OTMA destination routing descriptor, which contains the destination tpipe name, adapter name, and converter. For synchronous call-out, the use of OTMA routing descriptor is required so that only one descriptor definition is associated with each call-out destination. See Example 6-33. The call-out request message is queued in this tpipe.

   Example 6-33 Sample OTMA Descriptor for call-out to IMS SOAP Gateway

   ```
   D SGDEST1 TYPE=IMSCON TMEMBER=HWS1 TPIPE=SGTP1 SYNTIMER=5000
   D SGDEST1 ADAPTER=HWSXMLA0 CONVERTR=XMLCNV1D
   ```

4. If a call-out request message is available on the tpipe, in response to the IMS Resume TPIPE request (issued by the SOAP gateway) IMS Connect retrieves the call-out request message from OTMA and checks the OTMA header.

   Each descriptor carries the TMEMBER and TPIPE name that indicate which tpipe the call-out message will be going to. It also contains the XML adapter and converter name that indicate which converter to be used for XML transformation for the call-out request message. If XML data transformation is used, the XML adapter name and the XML converter name, available with the OTMA routing descriptor are copied into the OTMA header during the OTMA routing descriptor processing.
5. In order to support IMS Synchronous call-out to Web Services, IMS SOAP Gateway implements the enhanced IMS Resume TPIPE protocol, and manages Resume TPIPE loop inside the IMS SOAP Gateway server to continually retrieve synchronous call-out messages that were inserted by the IMS application.
6. After the synchronous call-out request message is received, IMS SOAP Gateway processes the request, finds the Web service identifier in the call-out request message and matches it with the deployed Web Services to locate.
7. The desired Web Service is invoked. For synchronous call-out, IMS SOAP Gateway always expects a request-response operation. If the operation is one-way, it is considered an error scenario for IMS SOAP Gateway. If the operation is a request-response, IMS SOAP Gateway waits for the response data to return synchronously from the same SOAP/HTTP connection.
8. When IMS SOAP Gateway receives the response message, it first extracts the response payload data from the SOAP message. Then, an IMS Connect message is built to send the response message back to the originating IMS application through IMS Connect. IMS SOAP Gateway also sets a flag in the response message to indicate to IMS Connect that the response is for a synchronous call-out message. The message does not contain the transaction code for the IMS application (unlike the asynchronous call-out response handling). It contains the XML adapter name and the converter name if the user chooses to use the XML adapter function to transform the call-out response message. Both the XML adapter name and the converter name are obtained from the correlator file specified by the user during the deployment step.

After the response message is built, it is sent to IMS Connect using Commit Mode 0 with Send Only protocol.

9. When IMS Connect receives the XML response message from the IMS SOAP Gateway, the XML conversion function will convert the XML response data into the IMS application data format if the user chooses to use the XML adapter function. Upon getting the converted application data bytes from the converter, the XML adapter send the resulting response data message to the waiting IMS application.
10. After receiving the message the waiting application continues behind the ICAL.

Note: The IMS Connect XML adapter function can handle messages larger than 32 KB. It relieves the 32 KB segmentation limitation for synchronous call-out messages. No transaction code is needed when returning the call-out response from the external Web Service. Also on call-out, WS-Security can be used.

The starting point for the setup of the Web service, when IMS is the consumer, is the existing service, probably not in IMS. That means:

- The layout of the request/response is dictated by the existing service, and it will be difficult to match it against input/output messages in IMS with fields of fixed length. As a consequence, the role of a converter will be null or minimum, and the IMS program will have to deal with XML. Remember that parsers do exist and the JAXB technology is available.
- If you want to generate partial proxy code, knowledge of the WSDL interface is needed.

6.7.3 Summary

Call-outs can be asynchronous and synchronous.

Call-outs to stateless session beans (SLSB) and message-driven beans (MDB) require (as service producer) a J2EE environment (WebSphere Application Server). MDB is the easiest to implement.

The call-out to a Web Service, which uses IBM IMS Enterprise Suite SOAP Gateway offers the opportunity to access whatever service through the SOAP protocol. An important feature in IMS 11 is the availability of the Web Service Security.

7

Installation and migration considerations

This chapter provides information about the installation and migration tasks that you should consider if migrating from a supported IMS release to IMS 11.

This chapter contains the following topics:

- Prerequisites and coexistence
- Coexistence with IMS 11
- IMS library changes
- Installation Verification Program
- Syntax Checker
- Installation and migration tasks
- Review of migration considerations

7.1 Prerequisites and coexistence

This section describes the prerequisites, and coexistence for IMS 11. You may also refer to *IMS Version 11 Release Planning,* GC19-2442.

7.1.1 Preventive Service Planning

An important step before you start installing IMS 11, is to review the current Preventive Service Planning (PSP) information.

The PSP buckets maintain current lists (which have been identified since the package was created) of any recommended or required service for the installation of this package. This service includes software PSP information that contains HIPER and required PTFs against the base release.

If you obtained IMS 11 as part of a Custom-built Product Delivery Option (CBPDO), HOLDDATA is included.

If the CBPDO for IMS 11 is older than two weeks by the time you install the product materials, contact the IBM Support Center or use S/390® SoftwareXcel to obtain the latest PSP bucket information.

You can also obtain the latest PSP bucket information by going to the following Web site:

`http://techsupport.services.ibm.com/server/390.psp390`

For program support, go to the Software Support Web site:

`http://www.ibm.com/software/support/`

Upgrade and Subset

PSP buckets are identified by Upgrades, which specify product levels, and Subsets, which specify the FMIDs for a product level.

Table 7-1 shows the Upgrade and Subset values for IMS 11.

Table 7-1 IMS 11 PSP Upgrade and Subset ID

Upgrade	Subset	Description
IMS1100	HMK1100/GA	IMS V11.01.00 System Services
IMS1100	JMK1101/GA	IMS V11.01.00 Database Manager
IMS1100	JMK1102/GA	IMS V11.01.00 Transaction Manager
IMS1100	JMK1103/GA	IMS V11.01.00 Extended Terminal Option
IMS1100	JMK1104/GA	IMS V11.01.00 Recovery Level Tracking
IMS1100	JMK1105/GA	IMS V11.01.00 DB Level Tracking
IMS1100	JMK1106/GA	IMS V11.01.00 Java On Demand features
IMS1100	HIR2220/0830	Internal Resource Lock Manager (IRLM) V2.2

7.1.2 Support procedures

Table 7-2 identifies the function modification IDs (FMID) and component IDs (COMPID) for IMS 11 for use when you have to report any problems to your IBM Support Center.

Table 7-2 IMS 11 Component IDs

FMID	COMPID	Component name	RETAIN® release
HMK1100	5635A0200	System Services	100
JMK1101	5635A0200	Database Manager	101
JMK1102	5635A0200	Transaction Manager	102
JMK1103	5635A0200	Extended Terminal Option	103
JMK1104	5635A0200	Recovery Level Tracking	104
JMK1105	5635A0200	DB Level Tracking	105
JMK1106	5635A0200	IMS Java On Demand features	106
HIR2220	569516401	Internal Resource Lock Manager V02.02.00	220

7.1.3 Software configuration requirements

This section describes configuration requirements for major enhancements of this version. Software requirements outside IMS itself are listed in A.1, "IMS 11 packaging" on page 304. Many enhancements in IMS 11 require other components, which you might not have implemented, before you can take advantage of them. Table 7-3 lists the required components.

Table 7-3 Components required for IMS 11 enhancements

Enhancement	SCI	OM	RM	ODBM	IMS Connect
Database quiesce	Required	Required	Required-multi[a]	Optional	Optional
Open Database	Required	Recommended	Optional	Required	Recommended
User Exit	Required	Required	Optional	Optional	Optional

a. *Required-multi* means that the Resource Manager is required if more than one IMS is in the IMSplex.

If you are migrating from IMS 9, the IMS 10 enhancements in Table 7-4 also require components that you might not have implemented, before you can use them.

Table 7-4 Components required by enhancements new to IMS 9 users

Enhancement	SCI	OM	RM	ODBM	IMS Connect
ACB member OLC	Required	Required	Required-multi[a]	Optional	Optional
DRD	Required	Required	Optional	Optional	Optional
Parallel RECON access	Required	Optional	Optional	Optional	Optional
Sysplex serialized program management	Required	Required	Required with CF structure	Optional	Optional

a. *Required-multi* means that the Resource Manager is required if more than one IMS is in the IMSplex.

Fast Path 64-bit buffer manager

When running under z/OS 1.9, this buffer manager has additional overhead involved with managing private storage shared among multiple address spaces. From z/OS 1.10, this overhead is removed by the introduction of common 64-bit storage. Therefore, z/OS 1.10 or later is preferable.

7.2 Coexistence with IMS 11

There are restrictions and special compatibility considerations for coexistence of IMS 11 with earlier versions of IMS.

Many of these considerations only apply when migrating directly from IMS 9. For your convenience, we have labeled these instances as *IMS 9 only*.

7.2.1 Overview of coexistence APARs

Both IMS 9 and IMS 10 have coexistence APARs and related PTFs. Table A-3 on page 307 and Table A-4 on page 308 contain the coexistence APARs and PTFs that are required for various IMS functions.

For details, refer to *IMS Version 11 Release Planning,* GC19-2442.

7.2.2 General coexistence considerations

IMS 11 can coexist with previous versions, allowing existing applications and data to be used without change.

General coexistence considerations are as follows:

- You must build new application control blocks (ACBs) for all existing program specification blocks (PSBs) and database descriptions (DBDs).
- An ALL parameter that is specified at a system generation and a cold-start are required for online systems (DBCTL, DB/DC, DCCTL). All data sets must be formatted when IMS is initialized the first time.
- If you are installing multiple copies of IMS systems at different release levels in the same processor, the latest version of the IMS SVCs must be used by all IMSs.
- The IMS dump formatting module (DFSAFMD0)that is installed on the host z/OS system must be from the most recent IMS release.

  The Offline Dump Formatter from IMS 11, IMS 10, or IMS 9 works without modification, if the appropriate formatter libraries are used.
- You might have to make changes to programs that process the log because some log records that are created by database changes have been modified.
- You must restart and complete or otherwise recover all failed applications before migrating.

Note: You cannot use extended checkpoint to restart applications across different releases of IMS.

7.2.3 Abend dump formatting exit coexistence considerations

IMS 11 dynamically installs a new abend dump formatting exit module (DFSAFMX0) during IMS startup.

If you use earlier IMS releases, or use both IMS 11 and earlier IMS releases, you must still install the DFSAFMD0 module if you want to use IMS online dump formatting.

The DFSAFMD0 module must be the highest version prior to IMS 11. When all IMS systems (control region and batch regions) are IMS 11 or later, you can remove DFSAFMD0 from SYS1.LPALIB and from the IEAVADFM CSECT of z/OS module IGC0805A.

This IMS 11 enhancement is described in 2.8, “Dynamic Abend Dump Formatting exit” on page 70.

7.2.4 APPC local LU enhancements (IMS 9 only)

All members of an IMSplex that participate in APPC processing must be at an IMS 10 or later level before using the new APPCLLU startup parameter in the DFSDCxxx PROCLIB member.

Unpredictable results might occur if you attempt APPC processing in a mixed-version IMSplex because the lower than IMS 10 IMS systems always use the base LU for RACF security checking, even if the message originates in an IMS 10 that supports the APPCLLU parameter

7.2.5 Common Queue Server coexistence

The following general coexistence considerations exist for Common Queue Server (CQS):

- A control region at version “n” can register with an CQS at version n, n+1, or n+2.

  A control region cannot register with earlier CQS.
- A CQS at version “n” can connect to the same CF structure as a CQS at any of the following versions: n-2, n-1, n, n+1, n+2
- Any supported version of CQS can run on the same central processor complex (CPC).

7.2.6 DBRC coexistence considerations

An IMS 11 DBRC can coexist with an IMS 9 or an IMS 10 DBRC if you do the following tasks:

1. Apply the DBRC Migration and Coexistence SPEs to the IMS 9 or IMS 10 systems.
2. Upgrade your RECON data set to the IMS 11 format by issuing the following command:

   ```
   CHANGE.RECON UPGRADE
   ```

UPGRADE RECON

IMS 9 and IMS 10 RECONs may be upgraded to IMS 11 by executing the DBRC utility (DSPURX00) and using the **`CHANGE.RECON UPGRADE`** command with an IMS 11 SDFSRESL library.

After the RECON data set has been upgraded, the (small program enhancement) SPE allows DBRC to convert records to the appropriate release format depending on whether the record is being written to or the record is being read from the RECON data set. This approach allows IMS 9 or IMS 10 systems to use the RECONs after they have been upgraded to IMS 11.

The SPE does not, however, enable the down-level DBRC for the new functions delivered with DBRC in IMS 11.

Restriction: After a RECON data set has been upgraded to the IMS 11 level, it is not accessible to any pre-IMS 11 system that does not have the DBRC Coexistence SPE applied.

MINVERS

The MINVERS level must reflect the lowest level of IMS that uses or shares the RECON data sets.

Automatic RECON loss notification

If you are using automatic RECON loss-notification in a mixed-version environment, notifications from the IMS 10 or IMS 11 DBRC are sent to and processed by an IMS 9 DBRC, regardless of the DBRC group ID used by the IMS 10 or IMS 11 DBRC.

Notifications from an IMS 9 DBRC do not have a DBRC group ID associated with them, but an IMS 10 or an IMS 11 DBRC can process these notifications.

DBRC API coexistence considerations

A DBRC application that was compiled with version 1 of the DBRC API (introduced in IMS 9) can work with version 2 of the DBRC API (introduced in IMS 10), but cannot use any of the functions introduced with version 2 of the DBRC API.

Version 2 of the DBRC API supports the application in the same way as version 1 of the DBRC API.

Each DBRC API macro includes the VERSION= parameter. New functions, such as AUTH, and new options, such as READONLY=YES, require VERSION=2.0.

In IMS 9, the only valid value for VERSION was 1.0. It was also the default.
In IMS 10 and later, VERSION defaults to 2.0.

The IMS 10 enhancements that are in version 2 of the DBRC API are only available with IMS 10 and IMS 11. These enhancements can only be used by a DBRC application that has been compiled with version 2 of the DBRC API.

Parallel RECON Access coexistence considerations

Although an IMS 11 can coexist with earlier versions of IMS (with the proper SPEs applied), the Parallel RECON Access (PRA) function cannot be used until the following tasks are done:

1. All systems and jobs that access the RECON data set are migrated to at least IMS 10.
2. The value for MINVERS is set to a minimum of: `‘10.1’`

In fact, to implement PRA, you must do many other tasks that are outside the scope of this book. However, you may refer to *IBM IMS Version 10 Implementation Guide: A Technical Overview*, SG24-7526.

RECON time-stamp enhancements

Increased precision to the microsecond for all time stamps is provided in an IMS 10 RECON only when the MINVERS level is set to `‘10.1’` or later.

7.2.7 DLIModel utility

IMS 10 was the last release of IMS to support the z/OS-based batch DLIModel utility. It is not supported with IMS 11.

If you are using this function, you should migrate to the DLIModel utility Eclipse plug-in, which is part of the IBM IMS Enterprise Suite.

7.2.8 DRA enhancement coexistence considerations (IMS 9 only)

The IMS 10 and later DRA startup-table option PCBLOC=31 causes the PCB address list and the PCBs to be placed above the 16 MB line. AMODE 24 applications and applications running with IMS 9 cannot use this enhancement.

7.2.9 Dynamic resource definition coexistence considerations

The Destination Creation exit routine (DFSINSX0) supports an IMSplex where some systems are DRD-enabled and some are not. If the DFSINSX0 exit routine is set up to create runtime resource definitions (and the same exit is used across the IMSplex), the routing behavior differs depending on whether the master is DRD-enabled or not.

If you have already implemented DRD, IMS 11 will use your IMS 10 resource definition data sets (RDDS). When IMS 11 is cold-started, it can import its definitions from an RDDS created with IMS 10. Similarly, for a fallback from IMS 11 to IMS 10 you can cold-start the IMS 10 system and import definitions from an RDDS that was created by IMS 11. For both migration and fallback, the IMSID must be the same as that used by the system which created the RDDS.

7.2.10 Exit routine coexistence considerations

The Standard User Exit Parameter List (SXPL) that exit routines use changes with each version of the list. Exit routines that run in multiple versions of IMS must be sensitive to the version of the SXPL.

The user exit enhancements in IMS 11 introduce version 6 of the list (SXPLVER6).

Exit routines that run in IMS 9 or IMS 10 work without modification in IMS 11. They are not, however, able to use the functions included in IMS 11 without being modified.

The Data Entry Database Randomizing modules (DBFHDC40 and DBFHDC44) extended call interface (XCI) option was introduced in IMS 10. If, while in coexistence mode, your randomizing routine must run on earlier releases and if the routine references the version number field, the required maintenance must be applied to your earlier systems before the exit can reference that field.

The coexistence APAR/PTF for IMS 9 is PK40642/UK23974.

The IMS Connect exit parameter list (HWSEXPRM) is changed for IMS 11. You must reassemble and rebind the IMS Connect exit routines that use HWSEXPRM to pick up the changes.

TM and MSC Message Routing and Control User exit routines (DFSMSCE0) from IMS 9 and IMS 10 work without modification with IMS 11, but you must reassemble.

7.2.11 Fast Database Recovery coexistence considerations

A Fast Database Recovery (FDBR) region must be at the same release level as the IMS system it is tracking.

Since IMS 10, the FDBR region no longer requires MODBLKS DD statements (even if Dynamic Resource Definition is not enabled), and these definitions should be removed from the FDBR JCL. FDBR acquires MODBLKS definitions from the checkpoint log records.

7.2.12 Fast Path coexistence considerations

If you issue either of the following commands in a mixed IMSplex, the parameter OPTION(OPEN) is processed only on IMS 11 systems. The parameter is ignored on lower-version IMS systems.

```
UPDATE AREA START(ACCESS) OPTION(OPEN)
UPDATE DB AREA(*) START(ACCESS) OPTION(OPEN)
```

7.2.13 Online change coexistence considerations (IMS 9 only)

The Global Online Change coexistence SPE allows an IMS 9 to participate in Global Online Change with IMS 10 and later systems.

ACB library member online change

Any IMS that wants to use the ACB library member online-change function must be at IMS 10 or later. Additionally, the OLCSTAT must be initialized to the IMS 10 or later level.

Versioning of the OLCSTAT data set was introduced to IMS 10 with APAR/PTF PK37127/UK39556.

Dynamic resource definition

Global online change supports an IMSplex where DRD is enabled on some IMSs but not on others.

If the **`INITIATE OLC PHASE(PREPARE)`** command is specified with type MODBLKS alone, then IMSs with MODBLKS online change-enabled perform the MODBLKS online change and return a good completion code. IMSs with DRD enabled ignore the MODBLKS keyword and return a completion code of *not applicable*.

Resource consistency checking

IMS 11 and IMS 10 do not support the resource-consistency-checking function of the Resource Manager. The NORSCCC keyword in DFSCGxxx is still allowed for compatibility, but its values are ignored.

The resource-consistency-checking SPE allows IMS 9 systems to use this function among themselves while they are in an IMSplex with IMS 11 or IMS 10 systems.

7.2.14 Image copy coexistence considerations (IMS 10 only)

Image copies taken by using the fast-replication option (introduced in IMS 10) of the Database Image Copy 2 utility cannot be used by IMS 9 except for list or query processing.

7.2.15 IMS abend search and notification (IMS 10 only)

The IMS abend search and notification function, introduced in IMS 10, works only on IMS 10 and later systems.

7.2.16 IMS synchronous callout function coexistence considerations (IMS 10 only)

There are no coexistence considerations if an application program that uses the synchronous callout function in IMS 11 is used exclusively in IMS 11.

If you want to deploy an application program that uses the synchronous callout function in IMS 11 from an IMS 11 system to an IMS 10 system, you must enable the synchronous callout function in the IMS 10 system.

IMS 9 does not support the synchronous callout function.

To enable the synchronous callout function in IMS 10, refer to *IMS Version 11 Release Planning,* GC19-2442, which describes the required IMS and IMS Connect APARs/PTFs.

7.2.17 IMSplex coexistence considerations

This section lists coexistence considerations that apply to IMSplexes.

CSL coexistence

When you migrate a system that is using the Common Service Layer (CSL) address spaces, any address space may be migrated to IMS 11 before the other address spaces are migrated.

Having some SCI, RM, OM, and IMS subsystems on IMS 11 while others are on IMS 10 is permitted.

There are some restrictions with IMS 9. IMS 9 has certain coexistence SPEs that allow a mixture of IMS 9, IMS 10, and IMS 11 address spaces.

A good practice is to use IMS 11 CSL address spaces if any IMS subsystem in at IMS 11.

Systems management enhancements (IMS 9 only)

The system management enhancement coexistence SPE allows IMS 9 systems to coexist with IMS 10 or IMS 11.

IMS 9 systems that process transactions submitted from the OM (using the QUEUE TRAN command) will receive an AD[1] status code if they reply to the IOPCB.

Unsolicited message support (IMS 9 only)

The IMS 10 and later components can send their unsolicited messages to OM. Unsolicited message support and OM audit trail support is valid only for IMS 10 and later OMs.

[1] Function argument is not coded correctly.

7.2.18 IMS Connect support coexistence considerations

Considerations can apply when an IMS Connect from IMS 11 can coexist with an IMS Connect from IMS 9 or IMS 10.

IMS Connect in IMS 11 supports the assignment of data store connections to different super member queues in any version of IMS that supports OTMA super member queues.

An IMS 11 IMS Connect can coexist with an IMS Connect from IMS 10 or IMS 9 if you adhere to several considerations. To review the consideration, refer to *IMS Version 11 Release Planning,* GC19-2442.

If you plan to use either the purge-not-deliverable or reroute functions while coexisting with IMS 9, IMS OTMA co-requisite APARs are required. Refer to A.1.3.1, “Compatibility PTFs for IMS 9” on page 307.

7.2.19 IRLM coexistence considerations (IMS 9 only)

If you are running IMS 9 with IRLM 2.1, you must upgrade your IRLM to 2.2 before your IMS 9 system can coexist with IMS 11.

7.2.20 Java application coexistence considerations

Java applications that use the IMS-to-XML mapping rules available with IMS 9 can run without having to be modified in a mixed-version IMS environment where both IMS 9 and IMS 10 and later versions are present.

Java applications that use the extended IMS-to–XML mapping rules that are available with IMS 10 and later versions can run only where each IMS is at IMS 10 level or later.

To enable Java Dependent Regions to access DB2 Version 8 (program number 5625-DB2), APAR/PTF PQ74629/UQ77540 must be applied to DB2.

7.2.21 Large sequential data set coexistence considerations (IMS 9 only)

Large sequential data set support (introduced in IMS 10) cannot be used with IMS 9.

7.2.22 Log records

If you have application programs that process IMS log records, you should determine whether the programs are affected by the changes to the log records.

You can assemble DSECTs for IMS log records by using the ILOGREC macro.

Table 7-5 on page 237 shows the log records that are new or changed for IMS 11.

Table 7-5 New or changed log records for IMS 11

Type	Action	Description
X'11'	Changed	Start of Conversation
X'18'	Changed	Extended Checkpoint record
X'21	Changed	Database Close
X'221B'	New	CREATE, UPDATE, and DELETE commands for OTMADEST
X'30'	Changed	Message Prefix Change Record
X'4035'	New	Checkpoint record for OTMA descriptors
X'4081'	New	Checkpoint record for 64-bit Fast Path buffers
X'45'	Changed	Internal Resource Statistics
X'4515'	New	Statistics from the 64-bit storage manager
X'4516'	Changed	IMSFP 64-bit Buffer Manager Dependent Region Statistics
X'5945'	Changed	Dependent Region statistics
X'5960'	New	FP 64-bit subpool creation, expansion, or deletion
X'7029'	Changed	Online Change writes this record after PREPARE locks the OLCSTAT data set
X'7030'	Changed	Online Change writes this record before COMMIT locks the OLCSTAT data set
X'7031'	Changed	Online Change writes this record after COMMIT locks the OLCSTAT data set
X'7032'	Changed	Online Change writes this record after COMMIT or TERMINATE unlocks the OLCSTAT data set
X'9904'	Changed	Created by the logging option on the EXIT= parameter on the DBDGEN when changed data

The Fast Path log records are included in ILOGREC if you use ILOGREC RECID=ALL. For a specific Fast Path log record, use `ILOGREC RECID=59nn`. The old Fast Path log record macros, for example DBFLGSYN and DBFLSRT, are still available.

7.2.23 MSC coexistence considerations

IMS 10 and later systems support IMS Multiple Systems Coupling (MSC) and shared-queue networks of mixed IMS releases.

MSPLINK buffer size ranges were changed in IMS 10. The minimum size is 1024 bytes, maximum is 65536. (VTAM buffer sizes for IMS 10 and later can be any size and no longer have to follow an algorithm.) You need to validate that partner IMS systems have matching buffer specifications.

If the IMS 9 systems have link buffer sizes less than 1024 bytes, they must be modified to a minimum of 1024 to coexist with IMS 10 and later systems.

TM and MSC Message Routing and Control User exit routines (DFSMSCE0) from IMS 9 and IMS 10 work without modification with IMS 11, but you must reassemble.

The following exit routines work with an IMS 9 system, but are not supported with IMS 10 or later systems:

- Terminal Routing exit routine (DFSCMTR0)
- Input Message Routing exit routine (DFSNPRT0)
- Link Receive Routing exit routine (DFSCMLR0/DFSCMLR1)
- Program Routing exit routine (DFSCMPR0)

7.2.24 Open Database enhancements coexistence considerations

The ODBA interface from previous versions of IMS can coexist with IMS 11 without modification.

To use the Open Database enhancements in conjunction with an IMS 9 or IMS 10 system, you must apply the coexistence PTFs, which are listed in A.1.3, “Compatibility with other versions of IMS” on page 307.

ODBM uses the new ODBA CIMS CONNECT support. The SPE has supporting code in the IMS core ODBA modules that is required to allow an IMS 11 ODBM to communicate with an IMS 9 or IMS 10 IMS subsystem through ODBA.

The CIMS CONNECT call can also be used by existing or new ODBA application programs. For more information about CIMS CONNECT, refer to 3.8, “ODBA enhancements” on page 95.

The IMS Universal drivers can be used by distributed Java applications that access IMS 9 or IMS 10 databases; the IMS 9 or IMS 10 systems are part of a mixed-version IMSplex if:

- The plex contains an IMS 11 system (along with its ODBM and IMS Connect address spaces).
- the DRDA request is processed with an IMS 11 IMS Connect.
- The coexistence PTFs are applied.

7.2.25 OTMA coexistence considerations

When an IMS 11 OTMA coexists with earlier versions, the OTMA IMS 9 coexistence SPE is required if ALL of the following conditions are true:

- Shared queues is being used.
- The shared queues front-end IMS is IMS 10 or IMS 11 and the back-end IMS is IMS 9.
- Applications on the back-end IMS 9 system use the ALT-PCB to generate asynchronous output for OTMA clients.
- OTMA clients, such as WebSphere MQ or IMS Connect, connect to the back-end IMS 9 system to retrieve the asynchronous output.

Without the coexistence SPE, the OTMA client that is connected to the shared queues back-end IMS will fail to retrieve the ALT-PCB output messages.

If the OTMA transaction expiration function is activated by OTMA clients, you must ensure that the target IMS for the transaction is IMS 11. If the target is IMS 10 or earlier, the expiration request is ignored by IMS.

Note: OTMA transaction expiration is planned to be available for IMS 10 by service detailed in A.1.3, “Compatibility with other versions of IMS” on page 307.

7.2.26 RACF enhancements to replace SMU (IMS 9 only)

Enhancements to RACF to replace the SMU in IMS 9 require changes to the PROCLIB data set to disable resource consistency checking for the MATRIX library.

If a mixed-version IMSplex contains IMS 9 systems, you should specify the NORSCCC(MODBLKS) parameter in the DFSCGxxx member of the PROCLIB data set to disable the resource consistency checking for the MATRIX library.

7.2.27 Remote Site Recovery coexistence considerations

IMS 11 Remote Site Recovery (RSR) coexistence supports configurations in which:

- The tracking site TMS is at IMS 11 and the active site is at IMS 9 or IMS 10.
- The active site RECON data sets are at IMS 9 or IMS 10 with the IMS 11 DBRC coexistence SPE applied.
- The active TMS that is running ILS is at IMS 9 or IMS 10.
- One or more active IMSs are at IMS 9 or IMS 10.

IMS Version 11 Release Planning, GC19-2442 describes RSR-supported configurations.

7.2.28 Sysplex serialized program coexistence considerations (IMS 9 only)

Support for sysplex serialized program management is limited to IMS 10 and later systems. IMS 9 cannot use a Resource Manager to manage serial application programs.

If you have an IMSplex installation that includes IMS 9, you must use your original program management techniques in those IMS systems to prevent serial application programs from being scheduled in parallel.

7.2.29 Syntax Checker coexistence considerations

The IMS 11 Syntax Checker supports IMS 9, IMS 10, and IMS 11.Be sure that the version shown is correct when you use the Syntax Checker to check the parameters of earlier versions.

7.2.30 Type-2 QUERY and UPDATE command coexistence considerations

The type-2 commands QUERY TRAN and UPDATE TRAN no longer support certain status values as parameters; these values are no longer displayed as status. Table 7-6 lists the status parameters and the output fields are now displayed along with their values.

Table 7-6 Status, output field, value

Status	Output field	Value
CONV	CONV	Y
FPE	FP	E
FPP	FP	P
REMOTE	RMT	Y
RESP	RESP	Y

In an IMSplex that includes an IMS 9, to issue the QUERY TRAN command, specify the value by using both the STATUS filter (as was done before IMS 10) and one of the new filters: CONV(), FP(), RMT(), or RESP().

7.2.31 IMS utilities coexistence considerations

The Batch Backout, Log Archive, Log Recovery, Log Merge, and Log Analysis utilities function properly only when they process data that was created by an IMS subsystem or batch application program that is at the same release level as the utility program.

The IMS 11 Database Recovery (DFSURDB0) utility accepts log, image copy HISAM unload, and change accumulation records from IMS 9, IMS 10 or IMS 11.

The IMS 11 Database Change Accumulation (DFSUCUM0) utility accepts log and change accumulation records from IMS 9, IMS 10 or IMS 11.

Suggestions: A good practice is to:

- Use DBRC with all Database Change Accumulation and Database Recovery jobs, especially during migration and coexistence.
- Use the IMS 11 utilities when the input data for a DBDS contains log, image copy, or change accumulation records created by the IMS 11 system.

If you want to run JCL from previous versions of IMS with the IMS 10 and later change accumulation and time-stamp recovery utilities, the JCL might have to be changed because of the DBRC time-stamp enhancements that were introduced in IMS 10.

Restriction: Image copies created by an IMS 11 Database Image Copy 2 utility (DFSUDMT0) using the DFSMS fast replication copy option (FlashCopy® or Snapshot™) cannot be used as input to an IMS 9 Database Recovery utility.

7.2.32 IMS Tools migration and coexistence

The IBM IMS tools are specifically designed to enhance the performance and operation of IMS, and are upgraded and enhanced to work with IMS 11. Web link to the specific IMS 11 support information is listed at:

`http://www.ibm.com/support/docview.wss?rs=434&uid=swg21296180`

Some ISV products might require updates; contact your vendor for information.

7.3 IMS library changes

Changes to the IMS library for IMS 11 include:

- Minor title changes to all of the publications
- Merging four sets of publications
- A change to how IMS Messages and Codes are delivered
- A change to the number of formats that are provided

7.3.1 IMS library reorganization

The titles of all PDF publications in the IMS 11 library have been changed so that the publication titles match the navigation tree titles in the information center. Refer to *IMS 11 Release Planning,* GC19-2442 for an explanation of the changes and organization. Four sets of publications have been streamlined so that information is grouped in a more intuitive, and thereby more retrievable, organization.

Beginning with IMS 11, the four volumes of IMS Messages and Codes list messages and codes starting from IMS 10 and including all later versions. Individual messages and codes include version-specific changes, if applicable. Because the same messages and codes volumes support IMS 10 and later, those volumes are still be available in BookManager® format.

Table 7-7 summarizes the title changes.

Table 7-7 IMS 11 library compared to IMS 10

IMS 11 title	IMS 10 title or (titles)
Application Programming	Application Programming Planning Guide Application Programming Guide
Application Programming APIs	Application Programming API Reference
Command Reference, IMS Commands A-M	Command Reference, Volume 1
Command Reference, IMS Commands N-Z	Command Reference, Volume 2
Command Reference, IMS Components	Command Reference, Volume 3
Communications and Connections	Communications and Connections Guide
Database Administration	Database Administration Guide
Database Utilities	Database Utilities Reference
Diagnosis	Diagnosis Guide Diagnosis Reference
Exit Routines	Exit Routine Reference
Installation	Installation Verification Guide
Messages and Codes, DFS Messages	Messages and Codes, DFS Messages
Messages and Codes, non-DFS Messages	Messages and Codes, non-DFS Messages
Messages and Codes, IMS Abend Codes	Messages and Codes, IMS Abend Codes
Messages and Codes, IMS Component Codes	Messages and Codes, IMS Component Codes
Operations and Automation	Operations and Automation Guide
Release Planning	Release Planning Guide
System Administration	System Administration Guide IMSplex Administration Guide
System Definition	System Definition Guide System Definition Reference
System Programming APIs	System Programming API Reference
System Utilities	System Utilities Reference

7.3.2 IBM information centers

IBM strategy is to deliver product information in Web-based information centers. Therefore, the IMS 11 information is optimized for viewing in the information center (which also includes PDF versions of books) and, except for messages and codes, is not provided in BookManager format. Information centers provide advantages over PDF and BookManager formats.

You may search across the entire information center, or set filters to search categories of information (for example, search only IMS 11 information, or search only IMS 10 information). You may also search for information center content by using a search engine such as Google or from:

`http://www.ibm.com`

You can subscribe to information updates by using RSS feeds. Accessibility support is available. Support for languages other than US English is also available.

IMS 11 library in the Information Center

The Information Center has been updated to include information about IMS 11 and is available at the following Web site:

`http://publib.boulder.ibm.com/infocenter/imzic`

The Information Center groups the IMS 11 books into several categories:

- IMS overview
- Release planning for IMS
- Installing IMS, which contains:
  - Program directory
  - Installation guide
  - System definition
- IMS administration, which contains:
  - Communications and connections
  - Database administration
  - Operations and automation
  - System administration
- Programming for IMS, which contains:
  - Application programming
  - Application programming APIs
  - System programming APIs
- Troubleshooting for IMS, which contains:
  - Diagnosis
  - Messages and codes
- IMS reference information, which contains:
  - IMS commands
  - Exit routines
  - Database utilities
  - System Utilities
- IMS glossary
- IMS newsletters

7.4 Installation Verification Program

The Installation Verification Program (IVP) process provides materials that you can use as a guide for working with your own IMS systems. The IVP process includes:

- Data set allocation
- Post-installation activities on target libraries
- System definition activities
- SVC considerations
- Authorization considerations
- IMS system preparation activities
- IMS application preparation activities
- IMS system and application execution activities

7.4.1 Overview enhancements

The IVP is enhanced to support the Open Database enhancements and support the new BPE-based DBRC startup JCL.

In IMS 10 and later, the Variable Export utility can be directly accessed as an option from the IVP Phase Selection panel, which helps you to more easily import the IVP variables from a prior release of IMS.

The IVP dialogs are completely replaced when a new release of IMS is installed.

These enhancements are discussed in 2.3, "Installation verification procedure" on page 31.

7.4.2 Running the IMS IVP dialog

The IVP dialog contains procedures to complete the activation, customization, and testing for the sample IMS system.

The IVP dialog is documented in *IMS Version 11 Installation,* GC19-2438.

You should use it to complete and verify the implementation and testing of IMS 11 because it is an excellent educational tool for learning how to install and customize IMS.

7.4.3 Phases of the IVP process

The IVP process consists of four phases: initialization, variable-gathering, file-tailoring, and execution. This section also discusses IVP output.

Initialization phase

The initialization phase begins each time you start the IVP dialog or change an option or sub-option. The IVP is driven from a set of ISPF tables, which contain information about the variables, jobs, tasks, and the sequence of those jobs and tasks you have to execute.

This phase has the following processes:

- The table-merge process

  A table-merge is necessary the first time you run the IVP and any time you make a change to existing environment options or sub-options that had not previously been selected, and when the installation of service requires it.

The table-merge process populates a set of customized tables from the master tables with the IVP variables, jobs, and tasks that are necessary for you to run the IVP based on your selected environment option and sub-options.

- The copy-startup-variable process

  After the table-merge process is complete or bypassed, the dialog compares the startup variables with their corresponding table values.

  If the table value is different and has not been changed by a prior copy-startup-variables process or by the CHG action in the variable-gathering phase, the table value is updated with the startup value.

Variable-gathering phase

In this phase, you choose the options that are used to produce the jobs and tasks necessary in the subsequent phases of the IVP, such as file-tailoring.

File-tailoring phase

The file-tailoring phase uses the ISPF file-tailoring services to combine the variables from the variable-gathering phase with skeletons from SDFSSLIB to create members (JCL and other materials) in INSTALIB.

Execution phase

The execution phase guides you through the jobs and tasks that are necessary to complete the building and running of the IVP system based on options that you chose.

Only the jobs and tasks that are specific to the selections that you made during initialization are presented. The jobs and tasks are presented in the order in which they are to be performed.

IVP output

The IMS environments that you can select include BATCH, DBCTL, DB/DC, DB/DC with XRF, and DCCTL. Most of the major functions of IMS can be demonstrated and tested using the IVP system. The IVP builds a viable sample IMS system in a controlled manner which is verifiable and robust.

The IVP verifies that both the IMS product itself and subsequent maintenance have been successfully installed. It implements and verifies the z/OS and VTAM interfaces and the various functions and features you selected.

The IVP builds, integrates, and allows execution of sample IMS applications. It also assembles the database resource adapter (DRA) interface module, which is used by DBCTL and Open Database Access (ODBA).

7.4.4 Starting the IVP dialog

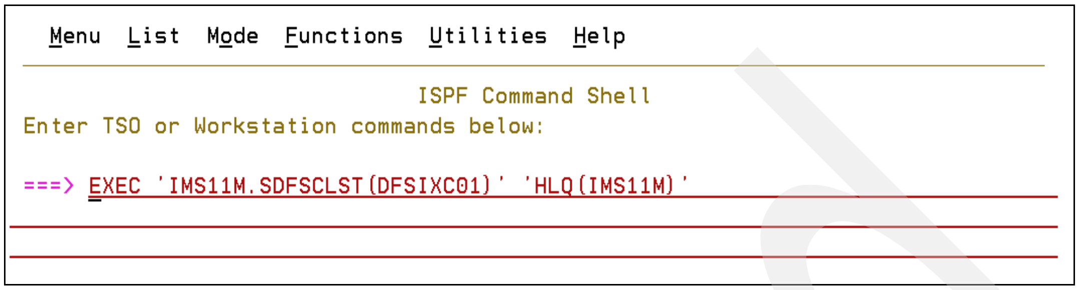
Start the IVP dialog by issuing an EXEC command from either an ISPF dialog or the IMS application menu.

Starting the IVP by using the EXEC command from within ISPF

You can start the IVP dialog from within ISPF either by using partial syntax with a simple command or by using full syntax.

To start the IVP dialog using partial syntax:

1. Open an ISPF application dialog.
2. Issue the TSO EXEC command (in the ISPF panel, option 6), shown in Figure 7-1.



```
  Menu  List  Mode  Functions  Utilities  Help
                              ISPF Command Shell
Enter TSO or Workstation commands below:

===> EXEC 'IMS11M.SDFSCLST(DFSIXC01)' 'HLQ(IMS11M)'
```

Figure 7-1 Command to start the IVP dialog from an ISPF panel

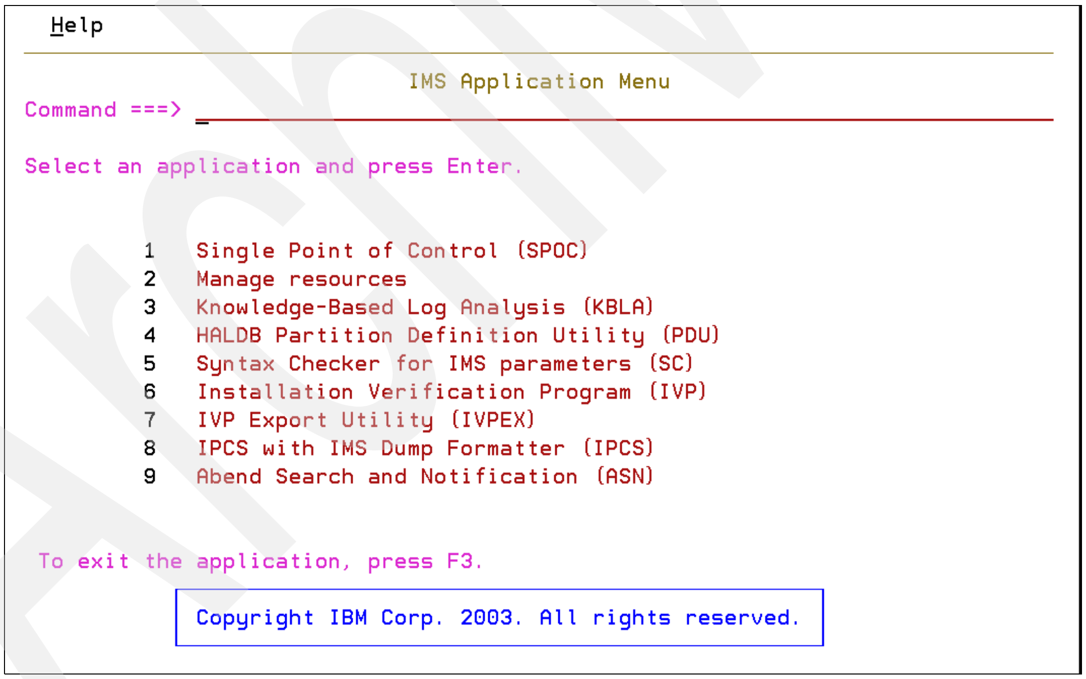
Starting the IVP from the IMS Application Menu

You can start the IVP dialog from the IMS Application Menu, as follows:

1. Open an ISPF application dialog.
2. Start the IMS Application Menu by issuing the following TSO EXEC command:

   ```
   EXEC 'sss.SDFSEXEC(DFSAPPL)' 'HLQ(qqq)'
   ```

   The IMS Application Menu opens. Figure 7-2 shows this panel.



```
  Help
                             IMS Application Menu
Command ===> _

Select an application and press Enter.


           1   Single Point of Control (SPOC)
           2   Manage resources
           3   Knowledge-Based Log Analysis (KBLA)
           4   HALDB Partition Definition Utility (PDU)
           5   Syntax Checker for IMS parameters (SC)
           6   Installation Verification Program (IVP)
           7   IVP Export Utility (IVPEX)
           8   IPCS with IMS Dump Formatter (IPCS)
           9   Abend Search and Notification (ASN)


 To exit the application, press F3.
             Copyright IBM Corp. 2003. All rights reserved.
```

Figure 7-2 IMS Application Menu

3. Select option **6** to start the IVP.

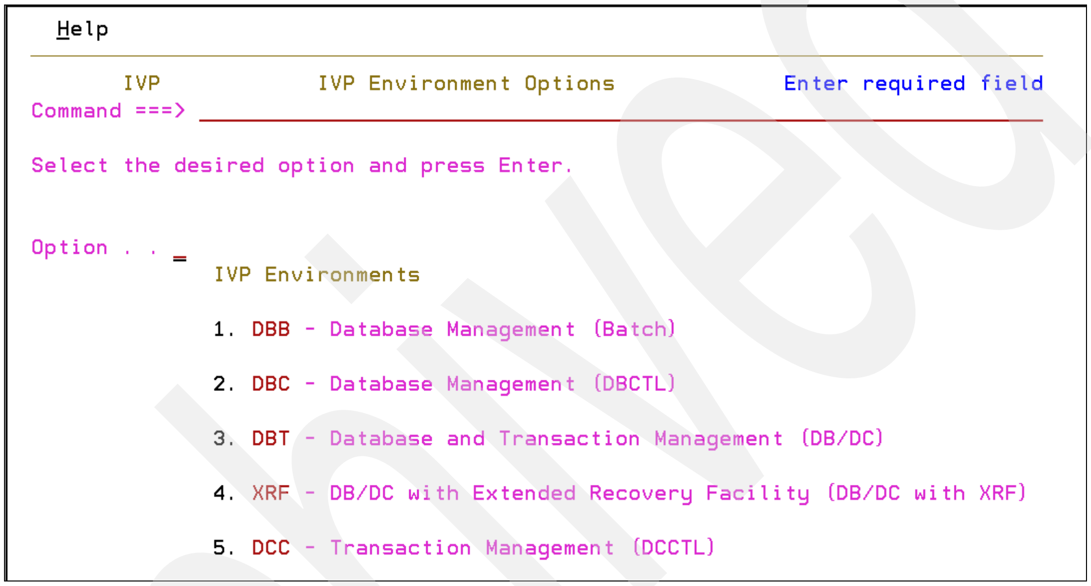
7.4.5 Starting the IVP initialization phase

During the IVP initialization phase, you select the installation option and sub-option values that the IVP uses to build customized tables of the specific jobs and tasks that have to be run.

Selecting the environment options

Select the options that apply to your environment. The IVP provides sub-options and tasks based on your choices to build a sample IMS system for installation verification.

Figure 7-3 shows the IVP Environment Options panel; this panel is referred to as the primary option menu for the IVP dialog.



```
  Help
 ------------------------------------------------------------------------------
       IVP           IVP Environment Options              Enter required field
 Command ===> ________________________________________________________________

 Select the desired option and press Enter.


 Option . . _
              IVP Environments

              1. DBB - Database Management (Batch)

              2. DBC - Database Management (DBCTL)

              3. DBT - Database and Transaction Management (DB/DC)

              4. XRF - DB/DC with Extended Recovery Facility (DB/DC with XRF)

              5. DCC - Transaction Management (DCCTL)
```

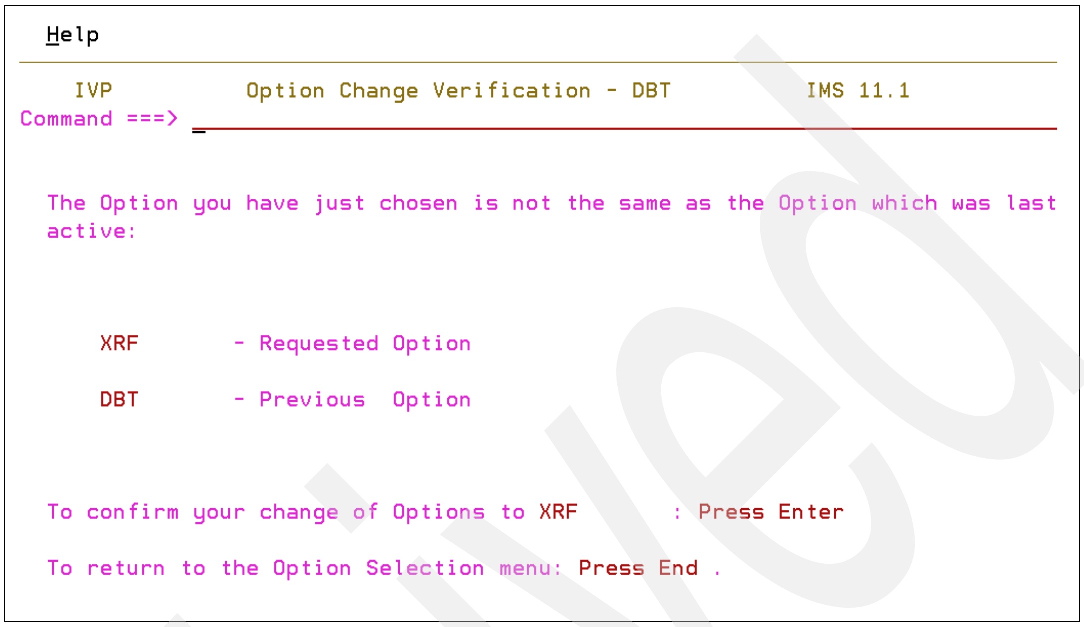
Figure 7-3 IVP Environment Options panel

If you previously ran the IVP dialog and made a selection in the IVP Environment Options panel, the Environment Option Change Verification panel opens.

If you did not previously run the IVP dialog, the Sub-Option Selection panel opens.

Verifying an environment option change

When you select an environment option that you did not select before, the Environment Option Change Verification panel opens. Figure 7-4 shows that the new option XRF is selected and that the last selected option was DBB.



```
  Help
------------------------------------------------------------------------
    IVP          Option Change Verification - DBT          IMS 11.1
Command ===> _


  The Option you have just chosen is not the same as the Option which was last
  active:


      XRF       - Requested Option

      DBT       - Previous  Option


  To confirm your change of Options to XRF       : Press Enter

  To return to the Option Selection menu: Press End .
```

Figure 7-4 Environment Option Change Verification panel

If the requested option change is correct, press Enter to confirm your selection.

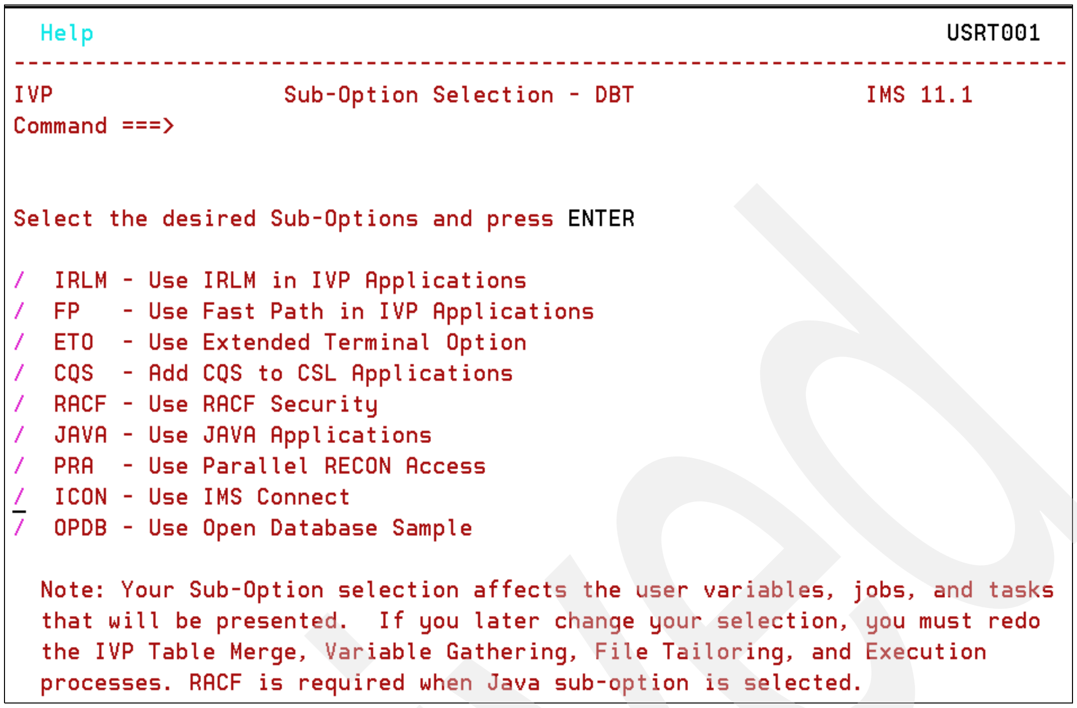
If the requested option is not correct, press End to return to the environment.

Selecting sub-options

Choose the sub-options that you want to add to your primary option selection, and these specify whether you want to use IRLM, Fast Path, and other IMS functions and features.

Ensure that the corresponding FMIDs for selected sub-options are installed during IMS product installation using SMP/E.

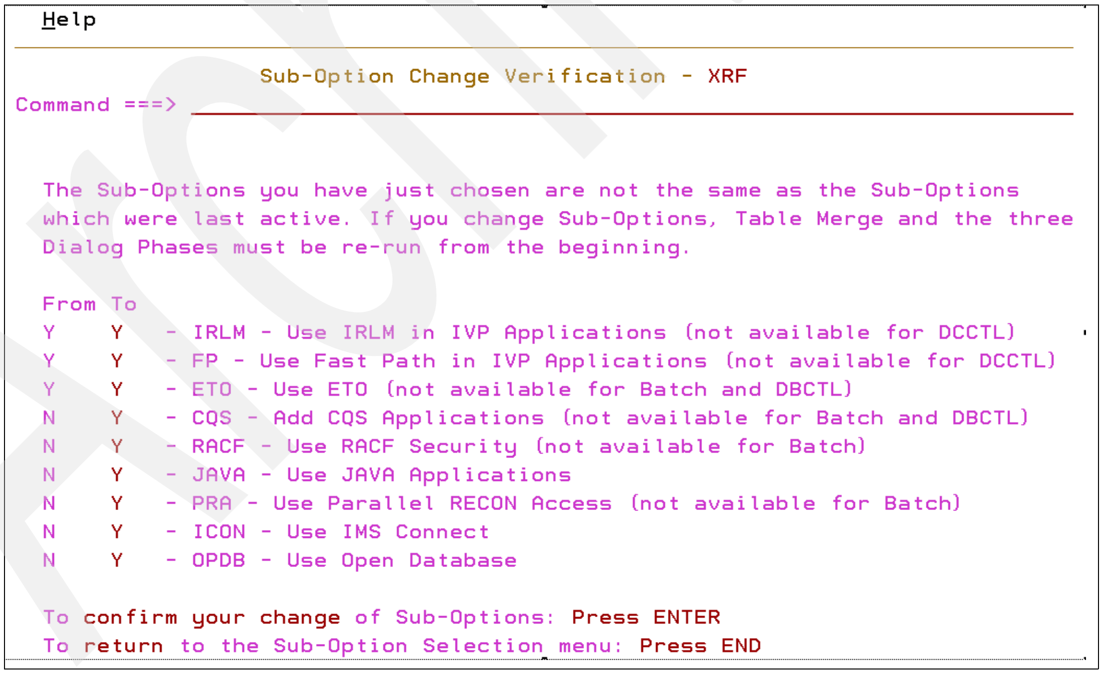
Figure 7-5 on page 248 depicts the IVP Sub-Options Selection panel of the IVP dialog.



```
  Help                                                                  USRT001
------------------------------------------------------------------------------
IVP                  Sub-Option Selection - DBT                  IMS 11.1
Command ===>


Select the desired Sub-Options and press ENTER

/  IRLM - Use IRLM in IVP Applications
/  FP   - Use Fast Path in IVP Applications
/  ETO  - Use Extended Terminal Option
/  CQS  - Add CQS to CSL Applications
/  RACF - Use RACF Security
/  JAVA - Use JAVA Applications
/  PRA  - Use Parallel RECON Access
/  ICON - Use IMS Connect
/  OPDB - Use Open Database Sample

  Note: Your Sub-Option selection affects the user variables, jobs, and tasks
  that will be presented.  If you later change your selection, you must redo
  the IVP Table Merge, Variable Gathering, File Tailoring, and Execution
  processes. RACF is required when Java sub-option is selected.
```

Figure 7-5 IVP Sub-Option Selection panel

If you change the selections that are displayed, the Sub-Option Change Verification panel opens. The dialog asks you to confirm your request for change. Figure 7-6 shows this panel.



```
  Help
------------------------------------------------------------------------------
                   Sub-Option Change Verification - XRF
Command ===> ________________________________________________________________


  The Sub-Options you have just chosen are not the same as the Sub-Options
  which were last active. If you change Sub-Options, Table Merge and the three
  Dialog Phases must be re-run from the beginning.

  From  To
  Y     Y    - IRLM - Use IRLM in IVP Applications (not available for DCCTL)
  Y     Y    - FP - Use Fast Path in IVP Applications (not available for DCCTL)
  Y     Y    - ETO - Use ETO (not available for Batch and DBCTL)
  N     Y    - CQS - Add CQS Applications (not available for Batch and DBCTL)
  N     Y    - RACF - Use RACF Security (not available for Batch)
  N     Y    - JAVA - Use JAVA Applications
  N     Y    - PRA - Use Parallel RECON Access (not available for Batch)
  N     Y    - ICON - Use IMS Connect
  N     Y    - OPDB - Use Open Database

  To confirm your change of Sub-Options: Press ENTER
  To return to the Sub-Option Selection menu: Press END
```

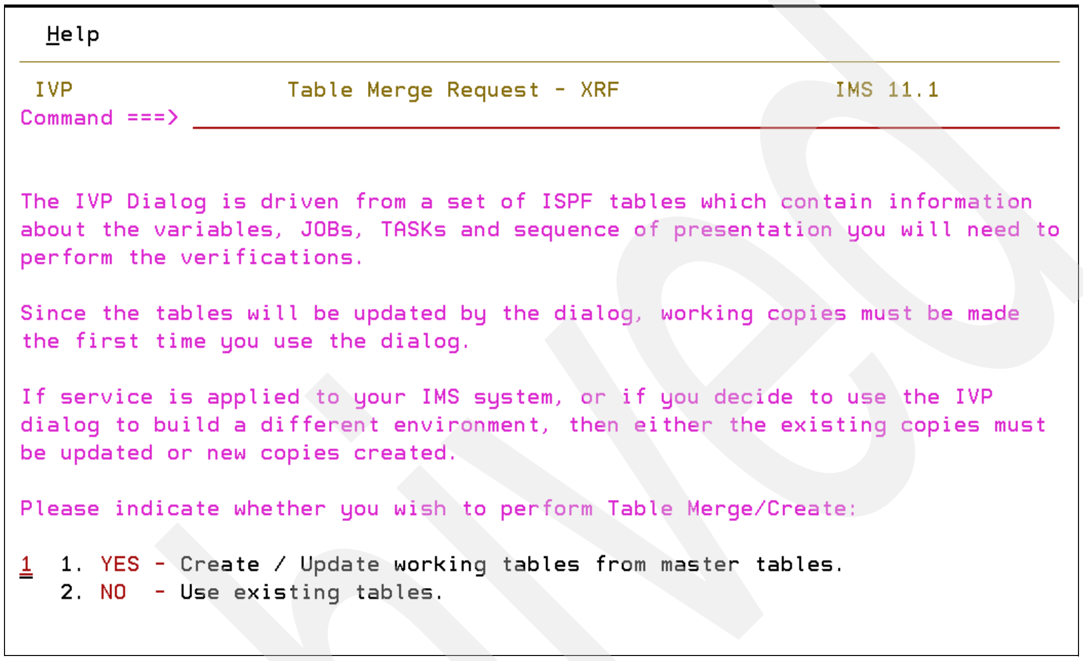
Figure 7-6 IVP Sub-Option Change Verification panel

If you are changing the selections after you have completed the table-merge, variable gathering, file-tailoring, or execution phases, you have to rerun the jobs and tasks in those phases.

Requesting a table-merge

After you select an environment option and sub-options, the IVP dialog gives you the option of performing a table-merge.

To request a table-merge, in the Table Merge Request panel showed in Figure 7-7, type 1 and press Enter. While the table-merge is in progress, the Table Merge in Progress panel opens and the keyboard is locked. This panel is updated as the tables are updated.



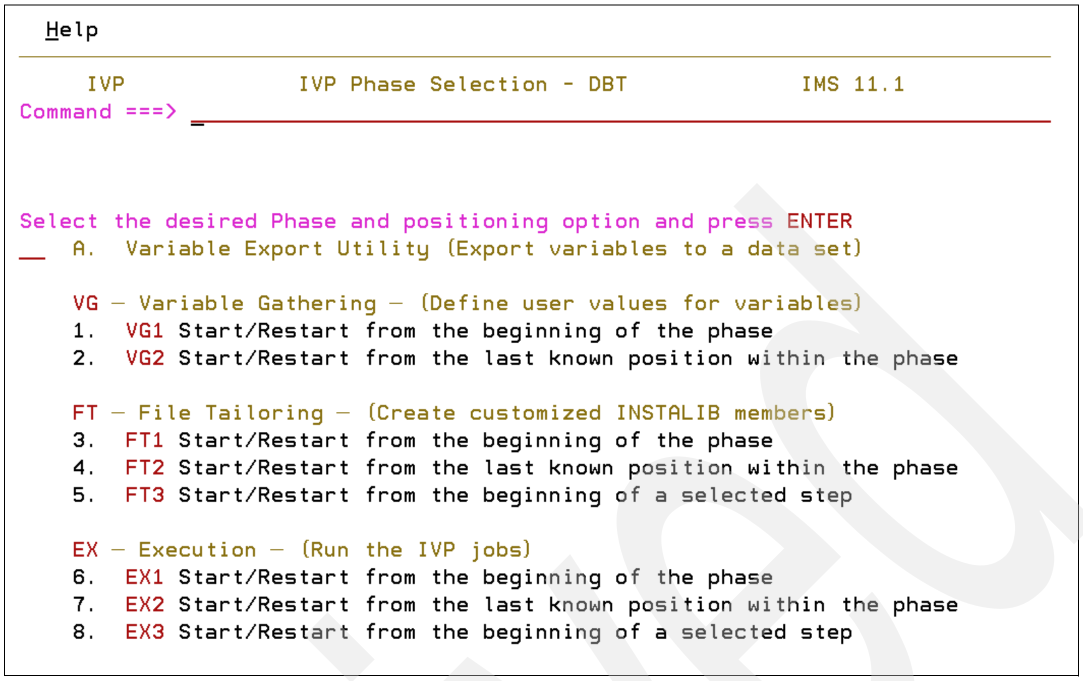
```
  Help
 ------------------------------------------------------------------------------
  IVP                   Table Merge Request - XRF                    IMS 11.1
 Command ===> ___________________________________________________________________


 The IVP Dialog is driven from a set of ISPF tables which contain information
 about the variables, JOBs, TASKs and sequence of presentation you will need to
 perform the verifications.

 Since the tables will be updated by the dialog, working copies must be made
 the first time you use the dialog.

 If service is applied to your IMS system, or if you decide to use the IVP
 dialog to build a different environment, then either the existing copies must
 be updated or new copies created.

 Please indicate whether you wish to perform Table Merge/Create:

 1  1. YES - Create / Update working tables from master tables.
    2. NO  - Use existing tables.
```

Figure 7-7 IVP Table Merge Request panel

After the table-merge process completes, the Table Merge Completed panel is displayed. Press Enter to continue.

Selecting an IVP phase and positioning option

Select an IVP phase and choose to start or restart from either the beginning of an IVP phase or from the last known location within a phase. Figure 7-8 shows the IVP Phase Selection panel of the IVP dialog.



```
  Help
 ------------------------------------------------------------------------------
     IVP            IVP Phase Selection - DBT              IMS 11.1
Command ===> _


Select the desired Phase and positioning option and press ENTER
__   A.  Variable Export Utility (Export variables to a data set)

     VG - Variable Gathering - (Define user values for variables)
     1.  VG1 Start/Restart from the beginning of the phase
     2.  VG2 Start/Restart from the last known position within the phase

     FT - File Tailoring - (Create customized INSTALIB members)
     3.  FT1 Start/Restart from the beginning of the phase
     4.  FT2 Start/Restart from the last known position within the phase
     5.  FT3 Start/Restart from the beginning of a selected step

     EX - Execution - (Run the IVP jobs)
     6.  EX1 Start/Restart from the beginning of the phase
     7.  EX2 Start/Restart from the last known position within the phase
     8.  EX3 Start/Restart from the beginning of a selected step
```

Figure 7-8   IVP Phase Selection panel

7.4.6 Gathering variables

Gathering variables involves making changes to prepare the JCL and other materials that are necessary for further customization in the file-tailoring phase.

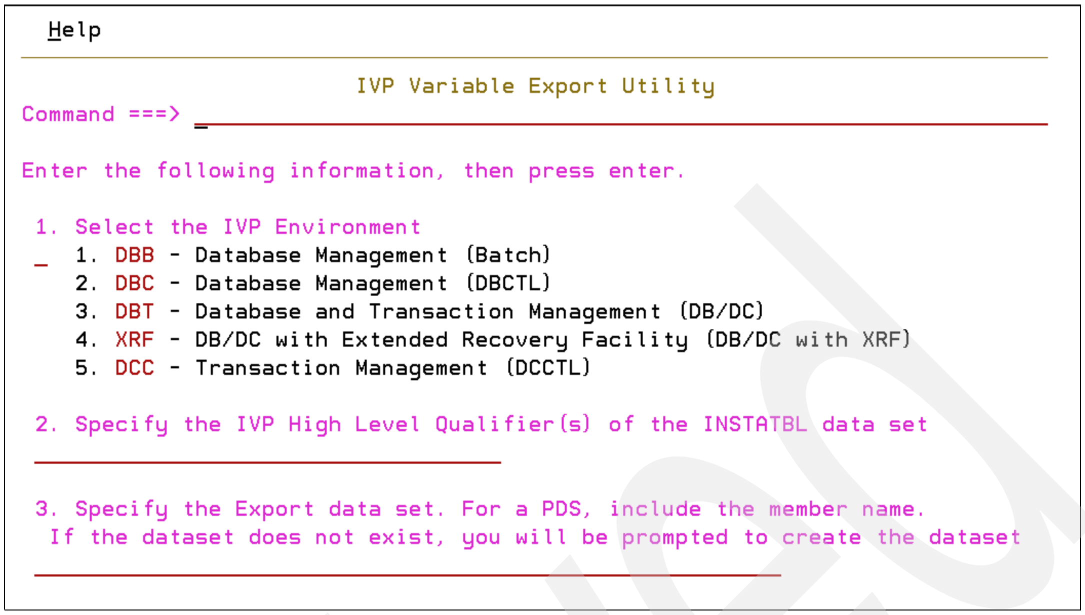
When you enter the variable-gathering phase, the IVP panel displays the variables based on your selections in the initialization phase. These variables are later used by the file-tailoring phase to customize the IVP to your environment and to create members in the INSTALIB data set.

You can export variables from an earlier iteration of the IVP dialog by using the IVP Variable Gathering Export and Import facilities.

Exporting and importing IVP variables

IVP variables can be optionally exported and imported between IMS releases or between different IVP dialog sessions of the same IMS release.

Use the IVP Variable Export utility, shown in Figure 7-9 on page 251 and that can be selected from the IMS Application Menu, to copy or export a set of previously used IVP variables to a sequential data set so you can subsequently import them.



```
  Help
 ------------------------------------------------------------------------------
                         IVP Variable Export Utility
Command ===> _____________________________________________________________

Enter the following information, then press enter.

 1. Select the IVP Environment
 _   1. DBB - Database Management (Batch)
     2. DBC - Database Management (DBCTL)
     3. DBT - Database and Transaction Management (DB/DC)
     4. XRF - DB/DC with Extended Recovery Facility (DB/DC with XRF)
     5. DCC - Transaction Management (DCCTL)

 2. Specify the IVP High Level Qualifier(s) of the INSTATBL data set

 ___________________________________

 3. Specify the Export data set. For a PDS, include the member name.
  If the dataset does not exist, you will be prompted to create the dataset

 ________________________________________________________
```

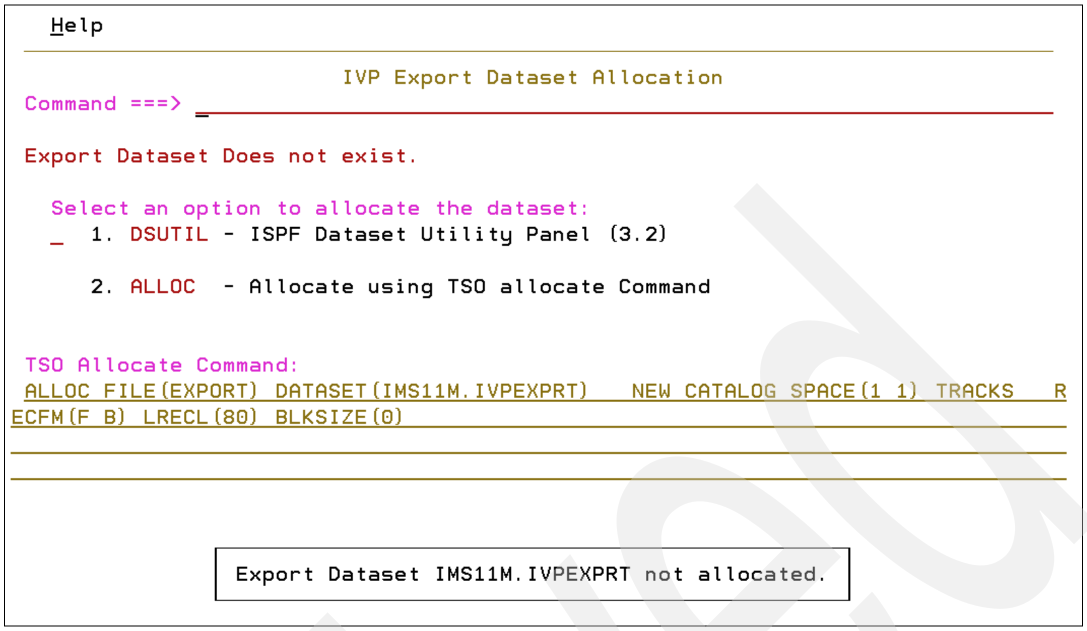
Figure 7-9 IVP Variable Export Utility panel

This data set can be subsequently imported to the IVP tables data set of the target IVP session.

You can use the ISPF split-screen capability to invoke the IVP Variable Export utility without exiting the IVP, as follows:

1. Open an ISPF application dialog.
2. Issue the following TSO command (where `HLQ` is the high-level qualifier of the IMS data sets) to launch the IVP Variable Export utility:

   ```
   EXEC 'hlq.SDFSEXEC(DFSIVPEX)' 'HLQ(hlq)
   ```
3. Provide the following information in the IVP Variable Export utility panel:
   - Select the environment option. Use the same option that you selected during the Initialization phase of the IVP process.
   - Select IVP high-level qualifier (HLQ), which identifies the IVP table data set (INSTATBL) from which you are exporting the variables.
   - Enter the export data set name in TSO data set format.
   - Press Enter to export the variables in the current IVP environment to the target IVP session.
4. If the export data set does not exist, the IVP Export Data Set Allocation panel opens as shown in the following Figure 7-10 on page 252.



```
  Help
                         IVP Export Dataset Allocation
Command ===> _

Export Dataset Does not exist.

  Select an option to allocate the dataset:
  _  1. DSUTIL - ISPF Dataset Utility Panel (3.2)

     2. ALLOC  - Allocate using TSO allocate Command


TSO Allocate Command:
 ALLOC FILE(EXPORT) DATASET(IMS11M.IVPEXPRT)   NEW CATALOG SPACE(1 1) TRACKS   R
ECFM(F B) LRECL(80) BLKSIZE(0)



                Export Dataset IMS11M.IVPEXPRT not allocated.
```

Figure 7-10 IVP Export Data Set Allocation panel

Select one of the following options to allocate the export data set:

– DSUTIL

The ISPF Utility Data Set Utility panel opens where you can specify the following attributes for the export data set:

DSORG	Sequential or partitioned
RECFM	FB
LRECL	80
BLKSIZE	27920 or another multiple of 80

– ALLOC

The utility executes the pre-build TSO ALLOCATE command you see on the panel. The data set name comes from the preceding panel.

Press PF3 or End to return to the IVP Variable Gathering panel

5. If the current IVP environment does not match the environment on which the variables were exported, the IVP Import Environment Mismatch panel opens. You can choose to continue the import process or cancel it.
6. Import the variables to the target IVP session from the export data set, as follows:
   a. In the Variable Gathering (LST mode) panel, issue the import action (**`Imp`**) command in the action field of any variable in the panel. This command imports all of the variables from an IVP export data set.
   b. Type the name of the export data set name in the TSO data set format. If the data set is a partitioned data set, include the member name.

Making global changes to variables

Use the export and import process during the variable-gathering phase to make global changes to variables by using the ISPF editor.

7.4.7 Tailoring files

In the file-tailoring phase, the IVP uses variables that you specified during the variable-gathering phase to prepare a customized set of IVP JCL and tasks to be stored as members of the INSTALIB data set for use in the execution phase.

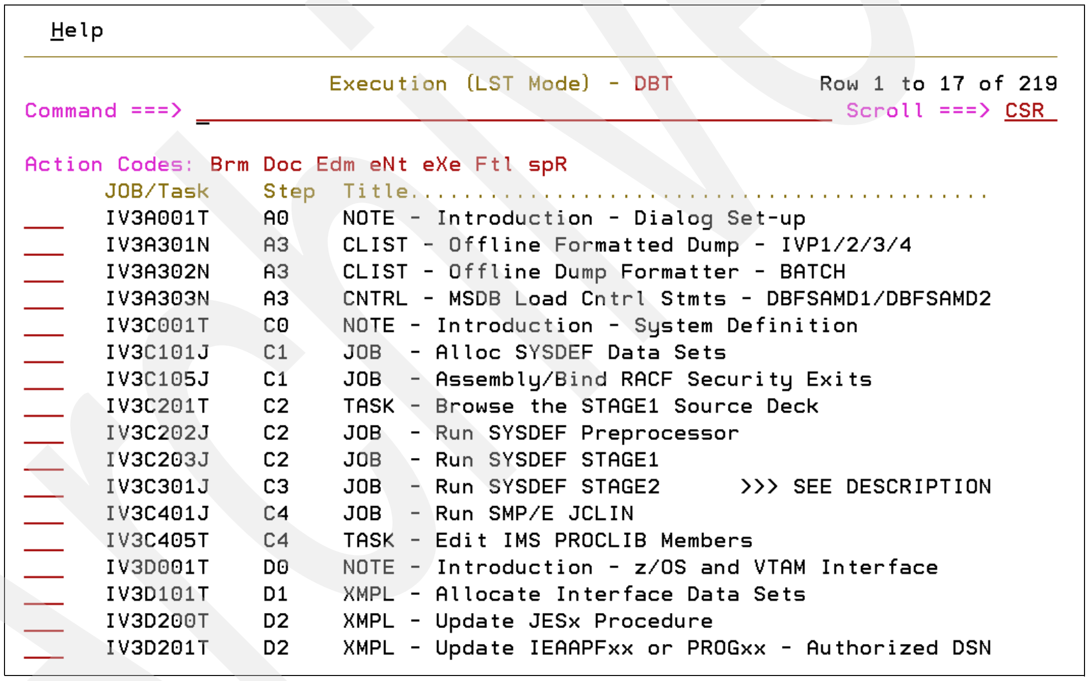
The ISPF file-tailoring facility creates this input by updating and building members in the INSTALIB data set based on the options you choose in this phase.

7.4.8 IVP jobs and tasks

The jobs and tasks that are actually presented in groups by the IVP dialog are determined by your choice of environment option and distribution media.

The last group listed, Steps Zx for index of additional PDS members, does not identify jobs or tasks in the IVP process. It identifies the members of DFSSLIB and DFSISRC libraries that support the IVP process.

Figure 7-11 shows a sample of the initials jobs and tasks presented by the IVP dialog.



```
  Help
 ------------------------------------------------------------------------------
                       Execution (LST Mode) - DBT              Row 1 to 17 of 219
Command ===> _                                                 Scroll ===> CSR

Action Codes: Brm Doc Edm eNt eXe Ftl spR
      JOB/Task    Step  Title.........................................
___   IV3A001T    A0    NOTE - Introduction - Dialog Set-up
___   IV3A301N    A3    CLIST - Offline Formatted Dump - IVP1/2/3/4
___   IV3A302N    A3    CLIST - Offline Dump Formatter - BATCH
___   IV3A303N    A3    CNTRL - MSDB Load Cntrl Stmts - DBFSAMD1/DBFSAMD2
___   IV3C001T    C0    NOTE - Introduction - System Definition
___   IV3C101J    C1    JOB  - Alloc SYSDEF Data Sets
___   IV3C105J    C1    JOB  - Assembly/Bind RACF Security Exits
___   IV3C201T    C2    TASK - Browse the STAGE1 Source Deck
___   IV3C202J    C2    JOB  - Run SYSDEF Preprocessor
___   IV3C203J    C2    JOB  - Run SYSDEF STAGE1
___   IV3C301J    C3    JOB  - Run SYSDEF STAGE2      >>> SEE DESCRIPTION
___   IV3C401J    C4    JOB  - Run SMP/E JCLIN
___   IV3C405T    C4    TASK - Edit IMS PROCLIB Members
___   IV3D001T    D0    NOTE - Introduction - z/OS and VTAM Interface
___   IV3D101T    D1    XMPL - Allocate Interface Data Sets
___   IV3D200T    D2    XMPL - Update JESx Procedure
___   IV3D201T    D2    XMPL - Update IEAAPFxx or PROGxx - Authorized DSN
```

Figure 7-11 Sample of initial jobs and tasks in the IVP dialog

Additional documentation for the IVP jobs, tasks, and variables can be printed using the DOC action during either the file-tailoring phase or the execution phase of the IVP dialog.

Use the IVP dialog to obtain current information regarding IVP jobs and tasks. In these lists, the jobs and tasks are presented in the same sequence that is used by the IVP dialog.

Table 7-8 lists the naming convention used for jobs and tasks presented as `IV_ssnnt`.

Table 7-8 IVP naming convention for jobs and tasks

IV_*ssnnt*	Meaning	Description
_	Underscore	The environment that you selected on the first panel: 1: DBB - Batch 2: DBC - DBCTL 3: DBT - DB/DC 4: XRF - DB/DC with XRF 5: DCC - DCCTL
ss	Step	IVP step
nn	Number	The number of the item within the step
t	Item type	J (JOB) A PDS member with the same name is placed into INSTALIB during the file-tailoring phase. Items of type J are intended to be submitted for execution. T (Task) Tasks represent items of work that must be prepared by the user. For some tasks, an example is provided in the INSTALIB data set. These examples are not intended for execution. N (Supporting materials) The INSTALIB data set can also contains members that support other jobs such as CLISTs and control statements.

7.4.9 Executing tailored jobs and tasks

You must process the jobs and tasks that were prepared by the file-tailoring phase individually through the execution phase.

To execute the IVP jobs and tasks, perform the following steps:

1. In the IVP Phase Selection panel, select option **6**, **7**, or **8**. Each selection within a phase provides a different positioning option.
2. Open each job and task. To view the instructions for each job and task, use the ENT action command.

   For IVP jobs you can browse, edit, or submit the job. Some items are nonexecutable examples, but the browse and edit actions are available to create an executable version of nonexecutable items.

   For IVP tasks, you are provided a scrollable description to assist you in performing the task.
3. Press End or PF3 when you are done.
4. Press Enter again if you completed the execution of all jobs and tasks, or press End to save your work if you want to complete the execution phase later.

7.4.10 Changes to the IVP for IMS 11

IMS 11 has added an IVP for Open Database.

Jobs and tasks before IVP Open Database

Before executing the IVP Open Database jobs and tasks, perform at least the following steps:

- Steps Cx for system definition (SYSDEF)

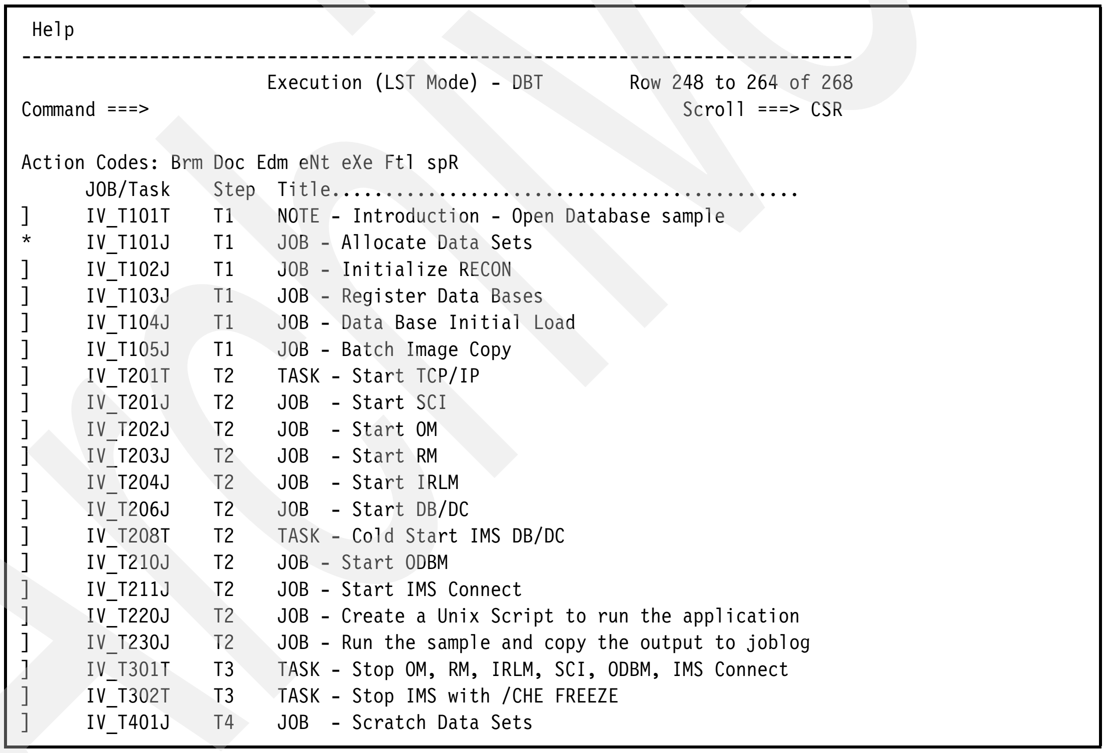
  The C series of steps include the jobs and tasks that are necessary to perform IMS system definition (SYSDEF).

- Steps Ex for preparing IVP applications and system

  The E series of steps include the jobs and tasks that you must perform to prepare the sample applications and the sample IMS system for execution.

IVP Open Database jobs and tasks

Figure 7-12 shows the jobs and tasks that appear in the IVP dialog and which the user must perform during the execution of Open Database sample. You should execute these steps in the sequence they are listed.



```
 Help
------------------------------------------------------------------------------
                       Execution (LST Mode) - DBT       Row 248 to 264 of 268
Command ===>                                                Scroll ===> CSR

Action Codes: Brm Doc Edm eNt eXe Ftl spR
      JOB/Task    Step  Title.........................................
]     IV_T101T    T1    NOTE - Introduction - Open Database sample
*     IV_T101J    T1    JOB - Allocate Data Sets
]     IV_T102J    T1    JOB - Initialize RECON
]     IV_T103J    T1    JOB - Register Data Bases
]     IV_T104J    T1    JOB - Data Base Initial Load
]     IV_T105J    T1    JOB - Batch Image Copy
]     IV_T201T    T2    TASK - Start TCP/IP
]     IV_T201J    T2    JOB  - Start SCI
]     IV_T202J    T2    JOB  - Start OM
]     IV_T203J    T2    JOB  - Start RM
]     IV_T204J    T2    JOB  - Start IRLM
]     IV_T206J    T2    JOB  - Start DB/DC
]     IV_T208T    T2    TASK - Cold Start IMS DB/DC
]     IV_T210J    T2    JOB - Start ODBM
]     IV_T211J    T2    JOB - Start IMS Connect
]     IV_T220J    T2    JOB - Create a Unix Script to run the application
]     IV_T230J    T2    JOB - Run the sample and copy the output to joblog
]     IV_T301T    T3    TASK - Stop OM, RM, IRLM, SCI, ODBM, IMS Connect
]     IV_T302T    T3    TASK - Stop IMS with /CHE FREEZE
]     IV_T401J    T4    JOB  - Scratch Data Sets
```

Figure 7-12 JOBs and TASKs of Open Database sample

Most of these tasks might be familiar to you if you have run an IVP before. Other steps are new, as follows:

- IV_T210J (start ODBM). An example of the ODBM JCL is in B.1.1, “ODBM JCL and configuration members” on page 315.
- IV_T220J creates a UNIX System Services script to run the IVP program, which is in Java.
- IV_T230J runs the script. The output is shown in Example 7-1.

If you are unfamiliar with IMS Connect, see the example included in B.1.2, “IMS Connect” on page 317.

Example 7-1 Open Database IVP output report

```
Host: 9.12.4.52 Port: 6000 Alias: IMOB User Name: ANDREWW
PSB for IVPDB1 Allocated
Batch Retrieved all segment instances of Phone Book
FIRSTNAME LASTNAME EXTENSION ZIPCODE
------------------------------------------------------------------
FIRST1  LAST1  8-111-1111 D01/R01
FIRST2  LAST2  8-111-2222 D02/R02
FIRST3  LAST3  8-111-3333 D03/R03
FIRST4  LAST4  8-111-4444 D04/R04
FIRST5  LAST5  8-111-5555 D05/R05
FIRST6  LAST6  8-111-6666 D06/R06
FIRST6  LAST6  8-111-6666 D06/R06

Inserted New Phone Book Entry with LASTNAME equal to LAST15

Batch Retrieved all segment instances of Phone Book and verify LAST15 was inser
FIRSTNAME LASTNAME EXTENSION ZIPCODE
------------------------------------------------------------------
FIRST1  LAST1  8-111-1111 D01/R01
FIRST15  LAST15  8-111-1515 D15/R15
FIRST2  LAST2  8-111-2222 D02/R02
FIRST3  LAST3  8-111-3333 D03/R03
FIRST4  LAST4  8-111-4444 D04/R04
FIRST5  LAST5  8-111-5555 D05/R05
FIRST6  LAST6  8-111-6666 D06/R06

Updated FIRSTNAME for segments with LASTNAME of LAST15

Batch Retrieved all segment instances of Phone Book and
verify that the FISTNAME was updated for the entry LAST15
FIRSTNAME LASTNAME EXTENSION ZIPCODE
------------------------------------------------------------------
------------------------------------------------------------------
FIRST1  LAST1  8-111-1111 D01/R01
NEWNAME  LAST15  8-111-1515 D15/R15
FIRST2  LAST2  8-111-2222 D02/R02
FIRST3  LAST3  8-111-3333 D03/R03
FIRST4  LAST4  8-111-4444 D04/R04
FIRST5  LAST5  8-111-5555 D05/R05
FIRST6  LAST6  8-111-6666 D06/R06

Segment with LASTNAME equal to LAST15 has been deleted

Batch Retrieved all segment instances of Phone Book and
verify that the segment with LASTNAME of LAST15 has been deleted
FIRSTNAME LASTNAME EXTENSION ZIPCODE
------------------------------------------------------------------
FIRST1  LAST1  8-111-1111 D01/R01
FIRST2  LAST2  8-111-2222 D02/R02
FIRST3  LAST3  8-111-3333 D03/R03
FIRST4  LAST4  8-111-4444 D04/R04
FIRST5  LAST5  8-111-5555 D05/R05
FIRST6  LAST6  8-111-6666 D06/R06
FIRST6  LAST6  8-111-6666 D06/R06

PSB Committed
PSB deallocated
Connection Closed
Open Database IVP Completed
```

If you want to run the Open Database IVP on a target system, to confirm that you have configured it properly, you must have:

- The sample database IVPDB1 (which you might already have)
- A copy of the UNIX System Services script
- A copy of the Java program (which you will already have if you have installed the IMS UNIX System Services components)

7.5 Syntax Checker

The Syntax Checker is an ISPF application that helps you define, verify, and validate parameters and their values in the members of the PROCLIB data set.

Online Help is available at the parameter level and provides assistance in moving to a new IMS release by identifying new parameters as well as any obsolete parameters from the previous release.

It saves the parameters to appropriate PROCLIB members in the correct format. The next time you access the changed control region, it starts with the new values.

7.5.1 Using the Syntax Checker

You can use the Syntax Checker to migrate your configuration members to IMS 11.

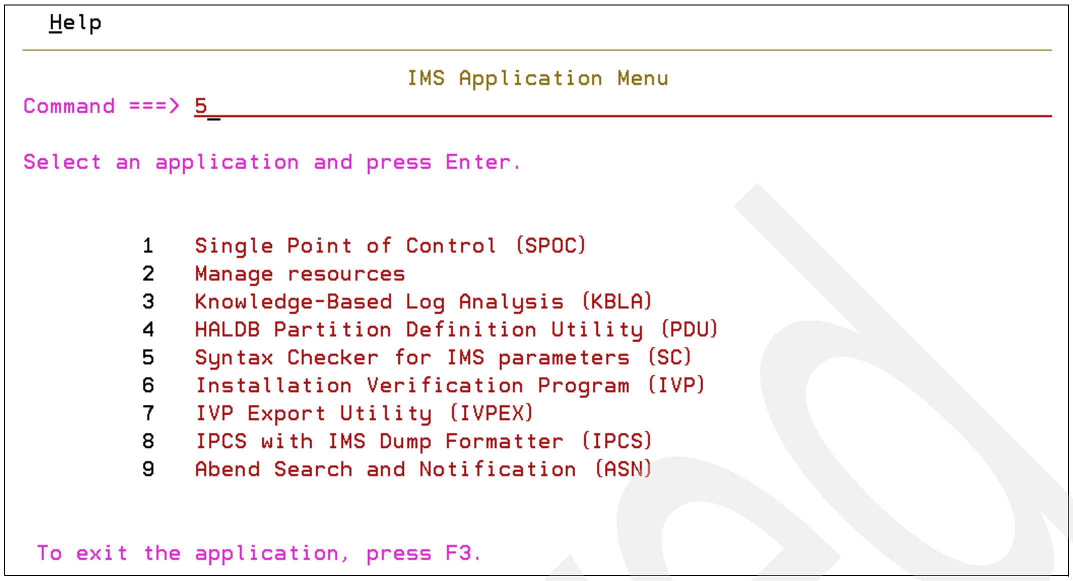
Starting the Syntax Checker

Start the IMS Syntax Checker by using either of the following two methods:

- Issue the following command in ISPF option 6, where HLQ is the high-level qualifier of the IMS data sets.

  ```
  EXEC 'hlq.SDFSEXEC(DFSSCSRT)' 'HLQ(hlq)
  ```

- Use the IMS Application Menu to select option **5**. It provides a common interface that run on TSO and ISPF and remains running until you exit it, manually. Figure 7-13 on page 258 shows this selection in the IMS Application Menu.



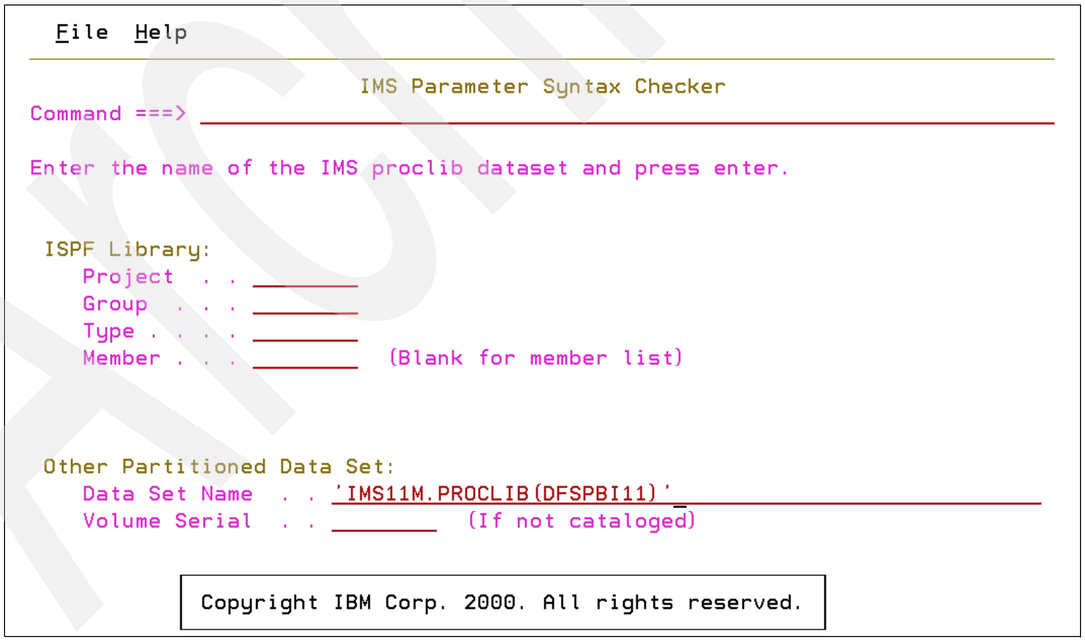
```
  Help
-------------------------------------------------------------------------------
                              IMS Application Menu
Command ===> 5_

Select an application and press Enter.


        1  Single Point of Control (SPOC)
        2  Manage resources
        3  Knowledge-Based Log Analysis (KBLA)
        4  HALDB Partition Definition Utility (PDU)
        5  Syntax Checker for IMS parameters (SC)
        6  Installation Verification Program (IVP)
        7  IVP Export Utility (IVPEX)
        8  IPCS with IMS Dump Formatter (IPCS)
        9  Abend Search and Notification (ASN)


 To exit the application, press F3.
```

Figure 7-13 IMS Application Menu

Using the Syntax Checker

After you start the Syntax Checker, the main panel (Syntax Checker Member and Data Set Name panel) shown in Figure 7-14 is displayed. Here, you can enter the PROCLIB data set name and the name of the member to be processed.

If you do not enter the member name, a member list is displayed. Select the member you want to process from the list.



```
  File  Help
-------------------------------------------------------------------------------
                         IMS Parameter Syntax Checker
Command ===> ____________________________________________________________________

Enter the name of the IMS proclib dataset and press enter.


 ISPF Library:
    Project  . .  ________
    Group  . . .  ________
    Type . . . .  ________
    Member . . .  ________  (Blank for member list)


 Other Partitioned Data Set:
    Data Set Name  . . 'IMS11M.PROCLIB(DFSPBI11)'_
    Volume Serial  . . ________  (If not cataloged)

             Copyright IBM Corp. 2000. All rights reserved.
```

Figure 7-14 Syntax Checker Member and Data Set Name panel

When you press Enter from the main panel, the Syntax Checker reads the input file and tries to determine the IMS release and type of control region.

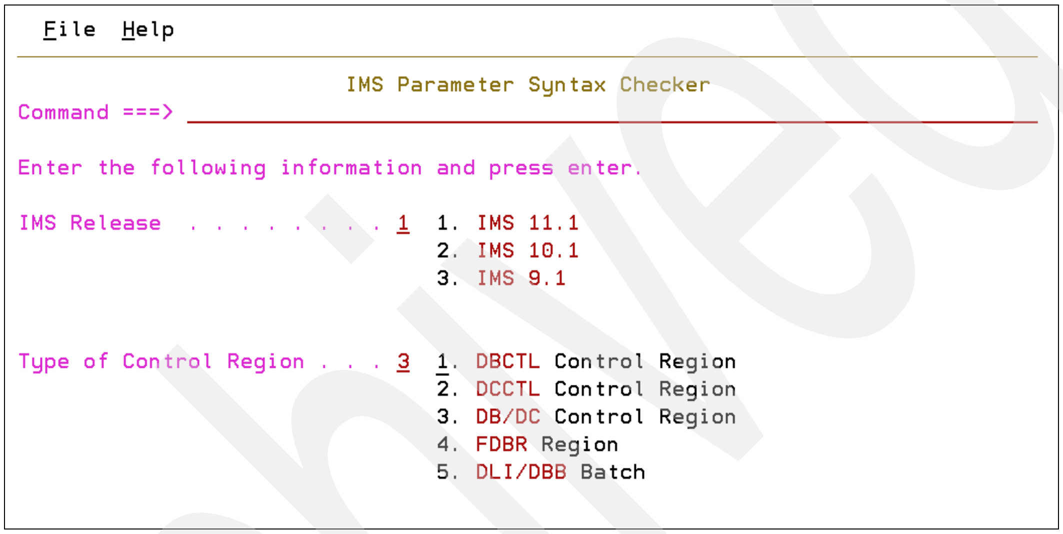
If the Syntax Checker cannot determine this information, one of the following panels opens:

- IMS Release and Control Region panel
- IMS Release panel

If the Syntax Checker is able to determine the information it requires from comments in the member, then the Syntax Checker Keyword Display panel is shown (refer to "Keyword Display panel" on page 260).

IMS Release and Control Region panel

The 'IMS Release and Control Region panel' showed in Figure 7-15 provides the Syntax Checker with IMS release and control region information that is required to process the member correctly.



```
  File  Help
 ------------------------------------------------------------------------
                         IMS Parameter Syntax Checker
Command ===> ____________________________________________________________

Enter the following information and press enter.

IMS Release  . . . . . . . . 1  1. IMS 11.1
                                2. IMS 10.1
                                3. IMS 9.1


Type of Control Region . . . 3  1. DBCTL Control Region
                                2. DCCTL Control Region
                                3. DB/DC Control Region
                                4. FDBR Region
                                5. DLI/DBB Batch
```

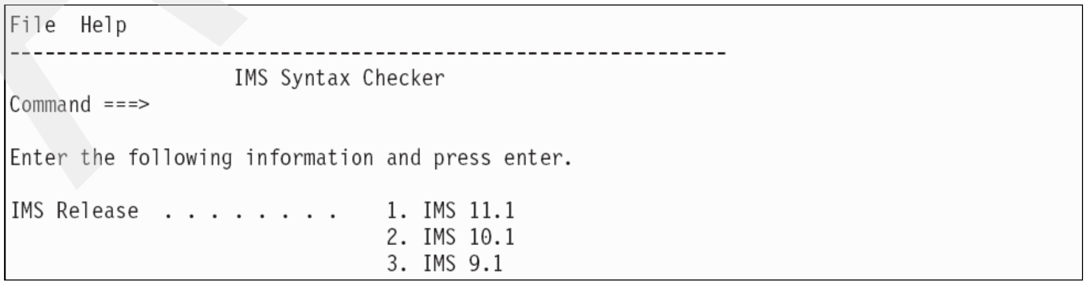
Figure 7-15 IMS Release and Control Region panel

When the Syntax Checker saves the member, it adds comment lines to the top of the member saving this information. The next time the Syntax Checker processes the member, this panel does not display.

After you type in the data and press Enter, the Syntax Checker Keyword Display panel opens.

IMS Release panel

The IMS Release panel, shown in Figure 7-16, prompts you to provide the Syntax Checker with IMS Release information that is required to process the member correctly.



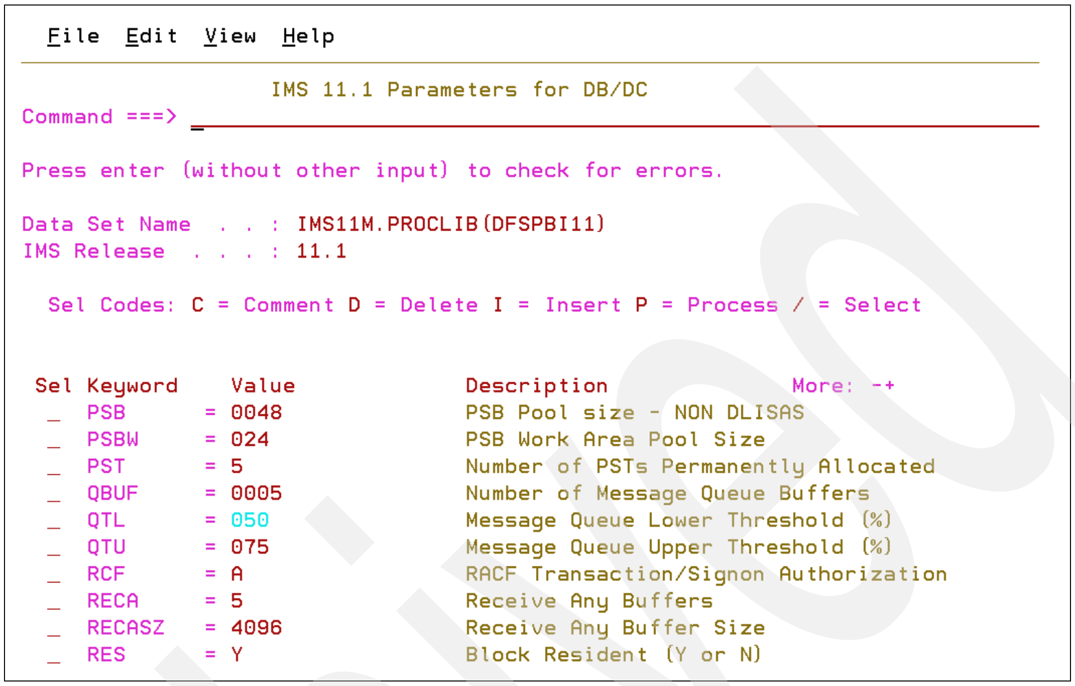
```
File  Help
-------------------------------------------------------------
                   IMS Syntax Checker
Command ===>

Enter the following information and press enter.

IMS Release  . . . . . . . .    1. IMS 11.1
                                2. IMS 10.1
                                3. IMS 9.1
```

Figure 7-16 IMS Release panel

After you type in the data and press Enter, the Syntax Checker Keyword Display panel opens.

Keyword Display panel

The Keyword Display panel, shown in Figure 7-17 of the IMS Syntax Checker, displays the keywords and their values, and indicates whether syntax errors exist.



```
  File  Edit  View  Help
 ─────────────────────────────────────────────────────────────────────────────
                    IMS 11.1 Parameters for DB/DC
Command ===> _

Press enter (without other input) to check for errors.

Data Set Name  . . : IMS11M.PROCLIB(DFSPBI11)
IMS Release  . . . : 11.1

  Sel Codes: C = Comment D = Delete I = Insert P = Process / = Select


 Sel Keyword    Value             Description              More:  -+
  _  PSB      = 0048              PSB Pool size - NON DLISAS
  _  PSBW     = 024               PSB Work Area Pool Size
  _  PST      = 5                 Number of PSTs Permanently Allocated
  _  QBUF     = 0005              Number of Message Queue Buffers
  _  QTL      = 050               Message Queue Lower Threshold (%)
  _  QTU      = 075               Message Queue Upper Threshold (%)
  _  RCF      = A                 RACF Transaction/Signon Authorization
  _  RECA     = 5                 Receive Any Buffers
  _  RECASZ   = 4096              Receive Any Buffer Size
  _  RES      = Y                 Block Resident (Y or N)
```

Figure 7-17 Keyword Display panel

If you select **Display new** from the View menu, the Syntax Checker lists only those parameters that are new in the IMS version you have selected.

To display the default values for parameters, press the F6 function key; press the key again to toggle between displaying and not displaying defaults. The default values are displayed in the description field of the parameter.

After modifying the member, press the Enter key without making any other modifications. Syntax-value checking is performed.

If errors exist, the first keyword with an error moves to the top of the display and an error message is displayed.

When all errors are resolved and you have modified the member as required, you can save the member to the originally selected PROCLIB and member or to a different PROCLIB and member.

Exiting the Syntax Checker

When you exit the Syntax Checker, it prompts you to save any unsaved changes.

7.6 Installation and migration tasks

This section is an overview of the tasks for migration to IMS 11.

7.6.1 Migration considerations

This section suggests general migration best practices for the following tasks of current IMS installations:

- Preparation tasks
- Installation tasks
- Validation tasks

Preparation tasks

Perform the following tasks in advance of the migration itself:

- Contact IBM Software Support for current installation, migration, and problem-resolution information; ask for PSP for IMS.

  Before you install IMS 11, check with your IBM Support Center or use either Information/Access or Service Link to determine whether additional preventive service planning (PSP) information is available that you must be aware of.

  The PSP upgrade name for IMS 11 is IMS1110.

- Read the *Program directory for Information Management System Transaction and Database Servers V11.01.00,* GI10-8788 for the most current hardware requirements, software requirements, prerequisites, and installation information.
- Review the IMS installation overview in *IMS Version 11 Installation,* GC19-2438.
- Review the service that has been applied to your current system. Determine if critical service has been included in the new IMS release. If not, acquire the appropriate service for the new IMS release.
- Review the functions and enhancements in *IMS Version 11 Release Planning,* GC19-2442. In particular, review changes to:
  - SMP/E, distribution, and system data sets
  - System definition macros
  - Log records
  - RECON records
  - Exit routines
  - Cataloged procedures
  - Control statement members in the PROCLIB data set
  - Utilities
  - Operator commands
  - Operating procedures
  - Messages and abend codes
- Review the z/OS interface considerations in *IMS Version 11 System Administration,* SC19-2443. It discusses SVCs and SYS1.PARMLIB updates.
- Install prerequisite software and maintenance.
- Determine the availability of updates to IBM IMS Tools, aids, and related products. A link to latest information is:

  lhttp://www.ibm.com/support/docview.wss?rs=434&uid=swg21296180

- Apply coexistence PTFs to your existing lower-level IMS systems that will operate with IMS 11. A list of these PTFs is in A.1.3, “Compatibility with other versions of IMS” on page 307.
- Examine Hardware Data Compression (HDC) dictionaries to determine if they incorporate IMS versions that are now out of service.

  Although rebinding dictionaries is not required when migrating to a new version of IMS, a gradual refresh of these dictionaries to a current release is a good practice.
- Evaluate and update IMS exit routines. If you use a RECON I/O Exit Routine (DSPCEXT0), you should examine it for required changes because of the change in RECON records.

  The DFSMSCE0 exit must be reassembled for use with IMS 11.

  Similarly, all IMS Connect exit routines must be reassembled when migrating IMS Connect to IMS 11. The IMS Connect HWSIMSO0 and HWSIMSO1 exit routines are not included with IMS 11. The HWSSMPL0 and HWSSMPL1 exit routines provide enhanced functionality and are delivered as source code with IMS 11 and previous versions. They should be used in place of HWSIMSO0 and HWSIMSO1.

Installation tasks

Perform the following tasks:

1. Install IMS 11 using SMP/E installation process.
2. Use the Custom-Built Product Delivery Offering (CBPDO) or ServerPac:
   - The CBPDO product package consists of one logical tape (multiple volumes). A CBPDO package that includes IMS can also include other products in the same System Release (SREL). CBPDO also provides service for the products included with the product order. The service includes all PTFs available within one week of order fulfillment. All PTFs are identified by one or more SOURCEIDs, including PUTyymm, RSUyymm, SMCREC, and SMCCOR.
   - ServerPac is a software delivery package. It consists of products and service for which IBM has performed the SMP/E installation steps and some of the post-SMP/E installation steps. To install the package on your system and complete the installation of the software it includes, use the CustomPac Installation Dialog, which is the same dialog used for all CustomPac offerings, including SystemPac® (dump-by-data-set format), ProductPac®, and RefreshPac.

     For IMS, ServerPac allocates, catalogs, and loads all the data sets; sets up the SMP/E environment; supplies a job to update PARMLIB (IEFSSNxx, PROGxx, IEASVCxx, and SCHEDxx); and directs you to start the IVP.
3. Validate your system definition source. You might want to merge the IVP source with your source.

   Run the IVP. The IMS IVP is used after the installation of a new IMS system, and is used to verify the installation, and can be used sporadically afterwards.

   The IVP Variable Export utility makes the migration of IVP variables between releases easier.

   Running the IVP is optional, but is a good practice. All required installation tasks are done outside the IVP. The IVP verifies that the installation is correct.
4. Install the system prerequisites and your new IMS system (including the pre-generation service).

   The complete set of IMS 11 modules that are necessary for execution are built by a combination of SMP/E processing and running an ALL-type of system definition process.

Most system definition statements from previous IMS releases are compatible with IMS 11.

If necessary, you can use the SMP/E GENERATE command to generate the JCL for jobs that will build the modules that are not built during the system definition process.

5. Install required service that was not included in the pre-generation service.
6. Install any required updates to IBM IMS and DB2 tools, aids, and related products.
7. Install the IMS 11 Type 2 and Type 4 SVC modules created by system generation.

   A z/OS IPL is not required. The SVCs can be installed by running DFSUSVC0 and specifying `SVCTYPE=(2,4)`.
8. Review the IMS Abend Formatting Module considerations in 2.8, "Dynamic Abend Dump Formatting exit" on page 70.

   IMS 11 installs the abend formatting routine (module DFSAFMX0) dynamically. No user setup is required to install DFSAFMD0 on the host z/OS system for IMS 11 as part of the IMS installation.
9. Upgrade the RECON data set, using the following command with the IMS 11 DBRC utility:

   ```
   CHANGE.RECON UPGRADE
   ```
10. Build application control blocks (ACBGEN).

Attention: Never use a higher-level ACB library with a lower-level IMS. You could cause an abend in the IMS control region or you could destroy some or all of your databases.

Validation tasks

Perform the following tasks:

- Validate users cataloged procedures.
- Validate user-created members of the PROCLIB data set.
- Validate, reassemble, and rebind exit routines and user modifications, especially IMS Connect exit routines and code that uses IMS control blocks, such as database randomizers.

  Check your exit routines before reassembling.
- Validate, reassemble, and rebind user programs that process log records.

  Some log record formats have changed.
- Validate and update operating procedures.

  Recovery, backup, and restart procedures
- Review the various execution parameters in the DFSPBxxx member of the PROCLIB data set that can affect performance and migration.

  Review and set the appropriate values for the AOIP, CMDP, DYNP, EMHB, FPWP, HIOP, LUMC, and LUMP parameters to specify an upper limit on the amount of storage a pool can acquire.

  You can also use the Syntax Checker to validate the values for the DFSPBxxx parameters.
- When using MSC to connect IMS systems with different releases, be sure to consider all message types (such as ISC, APPC, and OTMA) and the prefix sizes that accompany them. This is important.

  When message queue data sets are used, make the MSGQ LRECL and block sizes are identical across all IMS MSC systems.

A problem can occur when an IMS system is migrated to a new release that uses messages with larger prefix sizes and new prefix segment types.

Messages that contain these new and larger prefixes are sent to an earlier release of IMS; the new and larger prefixes might not fit the message queues of the earlier release of IMS.

- Consider other products that can be affected by migration. Any product that is dependent upon the format and contents of the IMS log or the RECON data set is potentially affected. Examples of affected products or utilities are:
  - IMS Statistical Analysis utility
  - IMS Fast Path Log Analysis utility
  - IMS Log Transaction Analysis utility
  - IMS MSC Log Merge utility
  - CICS®
  - Non-IBM products, including user modifications
- Stop your pre-version 11 system.
- Cold-start your IMS 11.
- Test your IMS 11 system.

7.6.2 Discontinuance of support

Support is discontinued for various utilities, exit routines, and functions, which are listed in this section.

IMS 9

IMS 9 marketing withdrawal of 6Sep2009, was announced on 2Jun2009 IMS 9 service discontinuance of 7Nov2010 was announced on 4Aug2009.

IMS 9 is the last version of IMS that supports:

- Security Maintenance utility (SMU)

  If you are using SMU, you should migrate to the RACF, or an equivalent product.

  For help, refer to 8.2, "Conversion utilities for SMU to RACF migration" on page 288.
- Basic Telecommunications Access Method (BTAM)

  If you are using BTAM, you should migrate to use VTAM or TCP/IP.

  User code or tools that depend on BTAM should migrate to VTAM or TCP/IP.

  If you are using VTAM Generic Resources (VGR) and are migrating Master Terminal Option (MTO) terminals from BTAM to VTAM, you might want to define the VTAM generic name on the GRSNAME= parameter in DFSPBxxx instead of using the following command:

  ```
  /STA VGR GRSNAME name
  ```

  MTO terminals are normally acquired automatically during IMS initialization when the VTAM ACB is opened.
- Terminal Routing exit (DFSCMTR0), Input Message Routing exit (DFSNPRT0), Link Receive exit (DFSCMLR0), and Program Routing exit (DFSCMPR0)

  These exits have been removed. You must use the IMS TM and MSC Message Routing and Control user exit (DFSMSCE0) as a replacement for the exits that are no longer supported.

  For information about the rules for calling the DFSMSCE0 user exit routine, refer to *IMS Version 11 Exit Routines,* SC19-2437.

- IMS Connect user message exits HWSIMSO0 and HWSIMSO1.

  If you use HWSIMSO0 and HWSIMSO1, you should migrate to use the sample user message exits HWSSMPL0 and HWSSMPL1.

  IMS Connect issues a warning message and ignores HWSIMSO0 and HWSIMSO1 if you specify them in the HWSCFGxx configuration member.

IMS 10

IMS 10 was the last release of IMS to support the z/OS-based batch DLIModel utility. It is not supported with IMS 11. If you are using this function, you should migrate to the DLIModel utility Eclipse plug-in which is part of the IBM IMS Enterprise Suite.

The JCA 1.0 resource adapter, one of the Java connectors in the IMS DB distributed resource adapter, is stabilized and is no longer being enhanced. You should switch to using the IMS Universal DB resource adapter that is delivered in IMS 11.

IBM has discontinued support for Enterprise Workload Manager™ (EWLM), so IMS can no longer offer this support. IBM is providing a transition for EWLM 2.1 customers to an IBM STG Lab Services offering. This new offering is designed to provide enhanced capabilities over the EWLM 2.1 product.

IMS 10 is the final version of IMS in which the IMS information is delivered in BookManager format.

7.6.3 Fallback considerations

IMS does not generally support downward compatibility in any major function between releases.

Consider the following steps when preparing your migration fallback plan. This information is intended as a guide to understanding fallback inhibitors, and should not be considered complete.

For each IMS that you are falling back, complete the following steps:

1. Ensure that the status of all databases updated by IMS 11 is correct.

   Establish a new recovery point for these databases by image copying them prior to allowing updates in the fallback release.

2. Resolve DBRC issues. Refer to 7.6.4, “DBRC fallback considerations” on page 266.

   Make sure you have the correct DBRC Coexistence PTF applied to the older IMS.

3. Shut down IMS 11.
4. Install the old version of IMS.

   You must rebuild your ACB library at the IMS version to which you are falling back.

   If global online change is enabled, ensure that the OLCSTAT data set is initialized to version 1. If the OLCSTAT data is initialized to a later version, and you are falling back to IMS 9 or earlier, the IMS initialization abends with abend U2800.

5. Cold-start the IMS.

You can use the IBM IMS Queue Control Facility for z/OS (QCF) to requeue IMS 11 messages to IMS 9 or IMS 10 message queues.

7.6.4 DBRC fallback considerations

Certain steps must be taken to revert DBRC to an IMS 9 or IMS 10 level.

BPE-based DBRC

If you have started to use BPE-based DBRC and have to fall back:

1. Shut down the IMS control region that is associated with the BPE-based DBRC address space.
2. Modify the DBRC procedure to use JCL appropriate for a non-BPE based DBRC region.
3. Restart IMS with DBRCNM referring to the non-BPE DBRC region startup JCL.

You might have to revert either or both of the following exits to older versions if you have changed these to *not* be compatible with the old calling interface:

- DSPCEXT0
- DSPDCAX0

Database Change Accumulation Utility (DFSUCUM0)

If you fall back from IMS 11 and you have change-accumulation data sets created by IMS 11, the logs in these data sets are not recoverable because the older utilities cannot process them.

Invalidate the IMS 11 change-accumulation data sets by running an image copy of the affected databases at the older level.

DBRC applications

As a rule, you should not modify your DBRC applications to use new functions until you are sure that you will not have to fall back.

If you must fall back from IMS 11 to IMS 9, and your system includes an application program that issues requests using version 2 of the DBRC API, the application must be reassembled using the IMS 9 SDFSMAC indicating that it now uses version 1 of the DBRC API.

After fall back, the application program is limited to using only the functions available with the DBRC API Version 1 in the IMS 9 system. If the application was compiled using version 1 of the DBRC API on IMS 11, no fallback action is required.

Lowering the minimum version value

You should not change MINVERS to `‘11.1’` until you are sure you do not have to fall back to a lower IMS version.

If however such a fallback becomes necessary, you can reduce MINVERS by following these steps:

1. Shutdown all IMS 11 subsystems.

   Confirm that IMS 11 subsystem records have been removed from the RECON data set. Use the **`LIST.SUBSYS`** command to see the subsystem records in the RECON data set. If IMS 11 subsystems remain in the RECON you should take the necessary recover actions.

2. Determine the status of your databases.

   If the databases need recovery with IMS 11 logs, perform those recoveries with IMS 11. If the databases need recoveries to a time prior to the IMS 11 updates, perform those recoveries with either the IMS 11 or the older version.

3. Use the following commands to delete all IMS 11 log records and all the allocation records on those logs from the RECON data set:

   ```
   DELETE.LOG
   DELETE.ALLOC
   ```

   Note: Time-stamp precision was increased to the microsecond level in IMS 10; what might appear as duplicate log records can result when falling back to IMS 9 and the time-stamp precision is reset to the tenth of a second.

4. If you are falling back to IMS 9 and had implemented Parallel RECON Access, you must disable it with the **CHANGE.RECON ACCESS(SERIAL)** command using IMS 11.
5. Reset the MINVERS value by issuing a **CHANGE.RECON MINVERS** command using IMS 11.

   If you receive message DSP1205E (meaning that the database quiesce flags are active), use the **CHANGE.DB** or **CHANGE.DBDS** command to set the flags off. After the flags are turned off, reissue the **CHANGE.RECON MINVERS** command.
6. Establish new recovery points for all databases updated by IMS 11 by taking image copies of those databases using the older version of IMS.

For details of the DBRC commands, refer to *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands* SC19-2432.

7.6.5 Dynamic resource definition fallback considerations

In general, you should not implement a complex function such as Dynamic Resource Definition (DRD) until you are sure you do not have to fall back.

If you must fall back to IMS 9 from an IMS 11 system that has DRD enabled:

1. Ensure that your MODBLKS data sets are current.

   You can do this by using one of the following methods:

   – Updating your stage-1 macros to reflect every change made with DRD and then running a MODBLKS gen.

   – Using the Resource Definition Data Set (RDDS) Extraction utility (DFSURDD0) to extract the resource definitions to create stage-1 macro statements from the stored resource definitions in an RDDS.
2. Shut down IMS normally.
3. Disable DRD in PROCLIB.

   Change the value of the MODBLKS keyword from DYN to OLC in DFSCGxxx or in the COMMON_SERVICE_LAYER section of the DFSDFxxx. Alternatively, you can remove the MODBLKS keyword because the default is OLC.

   If both members are defined, any values specified in DFSCGxxx override those in DFSDFxxx.
4. Ensure the Control Region JCL includes the MODBLKS DD statement.
5. Cold-start IMS.

   The online change process for MODBLKS is once again enabled. Variations of the dynamic resource definition CREATE, DELETE, and UPDATE commands that change definitions are no longer permitted.
6. Reinstitute your in-house procedures that use the online change process for the IMODBLKS data set and disable the procedures that use DRD commands.

7.6.6 Large sequential data set fallback considerations (IMS 9 only)

To fall back from using large sequential data sets:

1. Define data sets with less than 65 535 tracks on one volume.
2. Perform a database reorganization process (unload from the large sequential data sets and reload onto multiple volumes) if the large sequential data sets are used for databases.
3. Remove the JCL that defines the large sequential data set or sets.
4. Perform a cold-start to bring the multivolume data set or sets online.

7.7 Review of migration considerations

When migrating to a more recent release of IMS, be sure to review the release planning guides that span your specific migration path. These are:

- *IMS Version 11 Release Planning*, GC19-2442
- *IMS Version 10: Release Planning Guide*, GC18-9717
- *IMS Version 9: Release Planning Guide*, GC17-7831

Many of these considerations apply only when you migrate from IMS 9 directly to IMS 10. For your convenience we have marked these as *IMS 9 only*.

7.7.1 Migrating to IMS 11 Database Manager

Specific migration considerations apply when you are migrating from the IMS 9 or IMS 10 Database Manager to the IMS 11 Database Manager.

Database quiesce migration considerations

To use the database quiesce function, the MINVERS field in the RECON data sets must be set to: `‘11.1’`

DBRC API applications that interrogate the output from Query TYPE=DB, TYPE=DBDS, and TYPE=PART requests do not have to be modified if they do not want to access the fields added with the database quiesce enhancement.

Applications that want to get the new output must map to the new output fields and ensure that the DSPAPQHD block returned by DBRC has a minimum version of 3.0.

Enabling Fast Path (IMS 9 only)

As of IMS 10, the values on the FPCTRL macro are ignored, but the Fast Path default values remain the same. To enable Fast Path, define the new execution parameter FP=Y (the default value is FP=N) in DFSPBxxx and define the FPCTRL macro values as execution parameters.

If you specify FP=N (or do not specify anything), the Fast Path execution parameters are ignored and Fast Path is not enabled.

The FP= parameter can be changed only across a cold-start.

Leave the FPCTRL macro defined when you migrate, as IMS 11 ignores it. Then, if you have to fall back to IMS 9, the FPCTL macro is in place for use in the previous release. IMS 9 ignores the FP= execution parameter in DFSPBxxx.

SDEP Scan Utility exit, DBFUMSE1 (IMS 9 only)

The DEDB SDEP Scan utility exit routine is called by using AMODE 31, regardless of which binder AMODE specification you set, because the routine must be able to run in 31-bit addressing mode.

For more information about this routine, refer to *IMS Version 11 Exit Routines,* SC19-2437.

Fast Path 64-bit buffer manager

If your IMS installation did not use 64-bit storage prior to IMS 11, you have to make 64-bit storage available to use the Fast Path 64-bit buffer manager.

Fast Path usability and serviceability enhancements

If you have automated operations that are triggered when messages DFS2555I or DFS2716I are issued because no MSDBs are defined, you must modify them because these messages are not issued any more.

Main storage database terminals (IMS 9 only)

Verify that none of your main storage databases (MSDBs) are connected to Basic Telecommunications Access Method (BTAM) terminals because IMS 10 and later versions do not support BTAM terminals.I

Image copy enhancements (IMS 9 only)

If you do not want to use the fast replication enhancements, and column 63 and column 64 on the control statements are blank, you do not have to change existing JCL and existing skeletal JCL members.

As of IMS 10, the image copy type field in the IMAGE record in the RECON data set has moved and is expanded. Change any user or vendor processes that interrogate the IMAGE record to accommodate the moved and expanded image copy type field.

If you are migrating an Database Image Copy 2 utility (DFSUDMT0) job that does not use the fast replication enhancements that were introduced in IMS 10, you can enable your utility to take advantage of fast replication in IMS 11.

IRLM 2.2 (IMS 9 only)

Execution parameters have changed for IRLM 2.2.

You can run IRLM 2.2 with the same procedure that you use for IRLM 2.1 by changing the procedure to use the IRLM 2.2 RESLIB. None of the IRLM 2.1 parameters are rejected. However, the following execution parameters have changed for IRLM 2.2:

- PC=NO is ignored because IRLM 2.2 uses PC=YES exclusively.
- The MAXCSA value is ignored because it applies only to the PC=NO environment.
- The z/OS EXEC parameter MEMLIMIT has been added to the IRLM 2.2 startup procedure.

  If the value for MEMLIMIT is less than 2 GB. IRLM sets MEMLIMIT to 2 GB because IRLM 2.2 only accepts values of 2 GB or higher.
- The default for DEADLOK is (1,1).

Open Database enhancements migration considerations

You can migrate existing ODBA application servers to use the Open Database Manager (ODBM) delivered with IMS 11, and existing IMS DB resource adapter applications to use the new IMS Universal DB resource adapter, as follows:

1. Apply the appropriate coexistence APAR.
2. Add the IMSPLEX parameter and optionally the ODBMNAME parameter to the DFSPRP macro.
3. Reassemble and rebind the DFSxxxx0 load module (where xxxx is the DRA startup table name specified on the APSB call in the AIBRSNM2 field of the AIB).
4. After performing these tasks, you can simplify the ODBA applications by replacing multiple CIMS INIT commands with a single CIMS CONNECT command.

You can optionally migrate application programs that use the existing IMS DB resource adapter, which supports the JCA 1.0 architecture, to the new type-2 interface IMS Universal DB resource adapter, which supports the JCA 1.5 architecture.

WebSphere Application Server for z/OS applications that want to use the Open Database enhancements must deploy the new type-2 interface IMS Universal DB resource adapter.

After you have set up ODBM (refer to 3.1.2, “Setting up Open Database” on page 80), you can start to use the type-4 interface of the IMS Universal DB drivers.

7.7.2 Migrating to IMS 11 Transaction Manager

These topics describe the IMS considerations for migrating from IMS 9 or IMS 10 Transaction Manager to IMS 11 Transaction Manager.

APPC enhancements

Migrate all IMS systems that participate in a particular shared-queues environment to IMS 11 before attempting to use the APPC enhancements, even though the APPC local LU functions do not have any explicit migration considerations for IMS 11.

Expedited Message Handler (IMS 9 only)

When migrating to IMS 11 from IMS 9, you must modify existing automation programs and procedures that monitor static nodes or ETO dynamic users for input response mode to recognize a Fast Path input response mode status indicator, RESP-INP-FP.

Requeuing IMS messages (IMS 9 only)

If you use the IBM IMS Queue Control Facility for z/OS (5697-I08), apply APAR PK29667 for IMS 9 before you migrate to IMS 11.

MSC enhancements to the QUERY and UPDATE commands (IMS 9 only)

To use the MSC type-2 QUERY and UPDATE command parameters (introduced in IMS 10) with logical links, an MSLINK name is necessary.

You can define an MSLINK name in the new label field of the MSLINK system definition macro. If you do not specify an MSLINK name, IMS generates a default name of DFSLxxxx, (where xxxx is the logical link number generated by the system definition process).

OTMA migration considerations

Migration considerations for OTMA involve compatibility with the message control function, OTMA transaction monitoring, and IMS destination routing descriptors.

Message control function (IMS 9 only)

The message flood control function in IMS 10 and later is automatically enabled with a default limit of 5000 messages.

To provide compatibility with IMS 9 and deactivate the support, either specify an INPUT value of 0 in a descriptor or issue the **/STA TMEMBER INPUT** command.

The time-out support for synchronous CM1 message in IMS 10 and later is automatically enabled with a default value of 120 seconds.

To provide compatibility with IMS 9 and deactivate the support, specify time out value of 0 in a descriptor or issue the **/STA TMEMBER TIMEOUT** command.

OTMA transaction monitoring

IMS 11 can perform only limited OTMA transaction monitoring for OTMA clients that do not specify the transaction expiration time in the OTMA prefix or do not exploit the OTMA resources monitoring (such as IMS Connect in IMS 10 without the expiration PTFs).

In these cases, you can activate transaction expiration (but not message-level expiration) by specifying the EXPRTIME parameter in the TRANSACT macro or by issuing either of the following commands:

```
CREATE TRAN
UPDATE TRAN SET(EXPRTIME(seconds))
```

OTMA protocol messages regarding OTMA resource information are ignored by IMS Connect and other OTMA clients that have not taken advantage of OTMA resource monitoring.

IMS destination routing descriptors

In IMS 11, type-2 commands can take the place of destination routing descriptors, but both can coexist. These descriptors are defined in the DFSYDTx PROCLIB member and are processed when IMS starts.

In IMS 10, the definition of the destination routing descriptors in DFSYDTx must be entered from the most specific to the most generic destination routing descriptor name. Any destination with a masked character of asterisk (*) has to occur *after* the group of names that the asterisk is masking.

For example, following the order from most specific to most generic, you would have to list the contents in DFSYDTx as: DEST1234, DEST12*, and DEST*.

In IMS 11, this restriction is lifted and the user creating the descriptor entries no longer has to be aware of the order.

For example, the entries in DFSYDTx can be in any order, such as DEST*, DESTT1234, and DEST12*. OTMA automatically rearranges this internally from most specific to most generic.

PROCLIM changes (IMS 9 only)

For Fast Path potential transactions, the *processing limit time* (specified on the TRANSACT macro `PROCLIM=(count,CPU-time-per-transaction)` or through DRD) is treated differently in IMS 10 and later than it was in IMS 9.

IMS 9 evaluates the processing limit time for Fast Path potential transactions as:

- Seconds, if the transaction runs in an MPP region
- Hundredths of seconds, if the transaction runs in an IFP region

IMS 10 (or later) always evaluates the processing limit time Fast Path potential transactions as hundredths of seconds, regardless of whether the transaction runs in an MPP region or an IFP region.

This change might result in unexpected IMS 0240 abends when migrating from IMS 9 to IMS 11.

Evaluate the PROCLIM times for your Fast Path potential transactions and define a value that works for both MPP and IFP regions.

7.7.3 Migrating to IMS 11 system

These topics describe the IMS considerations for migrating from IMS 9 or IMS 10 systems to IMS 11 systems.

ACB library enhancements migration tasks

These topics discuss the migration tasks associated with the ACB library enhancements.

Caching ACBs into 64-bit storage

To migrate to ACBLIB members in 64-bit memory, complete the following steps:

1. Specify the ACBIN64 parameter in the DATABASE section of the DFSDFxxx PROCLIB member.
2. Stop IMS.
3. Restart IMS.

For more information, refer to 3.3.1, "Caching ACBs" on page 88.

Migrating ACB libraries to use dynamic allocation

To migrate ACB libraries to use dynamic allocation:

1. Create DFSMDA members for the ACBLIBA and ACBLIBB data sets.

   You can place DFSMDA members in a data set specified in either the IMS STEPLIB concatenation or the IMSDALIB concatenation. For more information, refer to 3.3.2, "Dynamic allocation of ACBLIBA/B" on page 89.
2. Remove the IMSACBA and IMSACBB DD statements from the IMS and DL/I JCL procedures.
3. Stop IMS.
4. Restart IMS with the DFSMDA members.

CQS migration

Migrate CQS and its control region (or regions) on the z/OS image at the same time. If this is not possible, CQS must be migrated before any of the control regions are migrated.

DBRC migration considerations

These topics describe the considerations and tasks for migrating DBRC to IMS 11.

RECON I/O exit DSPCEXT0 (IMS 9 only)

If you are migrating to IMS 11 from IMS 9, you must apply the migration/coexistence APARs to the RECON data set and evaluate your RECON I/O exit routine (DSPCEXT0), if you use one, because it might have to be modified to work correctly with the extended exit interface that was introduced in IMS 10.

BPE-based DBRC migration considerations

To start a BPE-based DBRC address space, create new DBRC JCL and update the IMS EXEC parameter, DBRCNM=, to specify the new member name.

If you are going to use your existing DBRC exit routines, there are no migration considerations.

If you want your DBRC exit routines to use the functionality of BPE, you must create new exits (which are based on current exits) that handle the BPE interface. For more information, refer to 5.1.2, "Exploiting BPE based DBRC" on page 149.

The new exits must have unique names and be added to the BPE user exit PROCLIB member.

Changes to the RECON data set

Certain records in the RECON data set are new or changed from the records in IMS 9 and IMS 10.

The following RECON records have new or changed fields as of IMS 10:

- RECON header: DSPRCNRC
- RECON header extension: DSPRCR1
- Image copy: DSPIMGRC
- Tracking subsystem: DSPSSRC

The following RECON records have increased in size and have new or changed fields as of IMS 11:

- DSPRCNRC: increased in size by 48 bytes
- DSPCHGRC: increased in size by 16 bytes
- DSPDBHRC: increased by 20 bytes

For detailed information about the RECON record types, refer to *IMS Version 11 Diagnosis,* GC19-2436.

Parallel RECON access

If you are migrating to IMS 11 from IMS 10 and Parallel RECON Access is in effect, you must ensure that there is no shunted I/O when the upgrade begins. The upgrade process begins with a quiesce close and a check for shunted I/O. The RECON data sets are closed and reopened in LSR mode. The records are upgraded as they are for non-PRA.

This includes upgrading the records in COPY1 and then upgrading the records COPY2. After the upgrade completes, the RECON data sets are reopened in PRA mode and the quiesce is ended.

For details about the RECON I/O exit routine (DSPCEXT0), refer to *IMS Version 11 Exit Routines,* SC19-2437.

Time-stamp precision migration considerations (IMS 9 only)

Change accumulation JCL and time-stamp recovery operations might have to be changed because of the increased time-stamp precision introduced in IMS 10.

IMS 9 JCL for change accumulation and time-stamp recovery operations might have to be changed because of the DBRC time-stamp enhancements that were introduced in IMS 10. Setting the MINVERS value from ‘9.1’ to a higher value changes time-stamp precision from tenths of a second to microsecond format.

After the MINVERS value is set to ‘10.1’ or later, certain DBRC commands require the use of the increased precision time stamp. For a list of the DBRC commands that are affected by this enhancement, see “DBRC time stamps” in the manual *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands,* SC19-2432.

Unconditional deletion of PRILOG information

If you plan to use this function, you should modify your DBRC security to include the new CLEANUP.RECON resource.

You might have to modify your procedures for maintaining the RECON data set in light of the CLEANUP.RECON command.

You might want to modify your automated programs or tools that issue DBRC commands.

Upgrading the RECON data set

To upgrade an IMS 9 or IMS 10 RECON data set:

1. Apply the IMS 11 coexistence SPEs to all IMS 9 and IMS 10 systems before you attempt to upgrade the RECON data set. These are listed in A.1, “IMS 11 packaging” on page 304.

   Jobs that access the RECON data set and do not create subsystem records, such as the Database Change Accumulation utility (DFSUCUM0) and the Database Recovery Control utility (DSPURX00), are not protected from having the RECON data set upgraded while they are running on a version of IMS that does not have the appropriate migration and coexistence SPE applied.

   When these types of jobs access the RECON data set after the upgrade, the results might be unpredictable. Make sure that no such jobs are running when you upgrade the RECON data set.

2. Make sure you have two active RECON data sets (COPY1 and COPY2) and a spare data set when you attempt to upgrade the RECON data sets while other jobs are accessing them.

3. Issue the **`CHANGE.RECON UPGRADE`** command. This command:

   – Upgrades the RECON data set without shutting down all IMS activity.

   – Uses the DBRC I/O recovery algorithms to recover from any failures during upgrade, relieving you of the necessity of making a backup of the RECON data set before upgrade and restoring it in case of a failure.

   – Must be issued using either the IMS 11 DBRC Recovery Control utility (DSPURX00) or the IMS 11 DBRC Command API request.

   If you use DBRC command authorization, consider setting the RECON qualifier as part of your migration process. You can set the RECON qualifier at the time of upgrade by adding CMDAUTH() parameters to the **`CHANGE.RECON UPGRADE`** command or after the RECON has been upgraded by issuing a **`CHANGE.RECON CMDAUTH()`** command.

   The **`CLEANUP.RECON`** command is a good reason why you should consider RECON command authorization. To see why, refer to 5.3, “Removing old data from the RECONs” on page 153.

4. When you are sure that a fallback to a previous IMS version is unlikely and all systems that access the RECON data set are IMS 11, update the minimum version value to: `‘11.1’`

You are not required to change the MINVERS value to `‘11.1’` if you have to use new function that requires this value.

After you set the MINVERS level for an IMS system, system signon fails for earlier versions of IMS for online environments. All other jobs accessing the RECON data set fail DBRC initialization if the version of IMS being used is less than the MINVERS level.

Dynamic resource definition migration considerations

If you are migrating from IMS 10 (with DRD enabled) to IMS 11, there are no migration considerations.

If you are migrating from IMS 9 or from IMS 10 without DRD enabled to a DRD-enabled IMS 11 system, the following steps describe one way to do this:

1. Shut down IMS normally.
2. Define the following DRD-related parameters:
   - Optionally, in DFSCGxxx, specify MODBLKS=DYN to enable DRD.
   - In the DYNAMIC_RESOURCES section of the DFSDFxxx member of the PROCLIB data set, specify:

MODBLKS=DYN	Use DYN if not already specified in DFSCGxxx.
RDDSDSN=	These data sets are a set of system definition data sets for the resource definitions.
AUTOEXPORT=AUTO	IMS exports definitions at each checkpoint if any definition has changed since the last checkpoint. AUTO is the default value.
AUTOIMPORT=AUTO	At cold-start, IMS imports resource definitions from the RDDSs if possible, otherwise from MODBLKS. AUTO is the default value.

3. Start IMS, specifying the execution parameter DFSDF=xxx, which identifies the resource definition member of the PROCLIB data set to use DFSDFxxx.
4. Cold-start IMS.

   Because the RDDSs are empty, IMS imports from MODBLKS. At the first checkpoint, IMS exports the definitions to the first RDDS.
5. Start using DRD.

Exit routine migration considerations

The user exit enhancements in IMS 11 introduce version 6 of the list (SXPLVER6). Exit routines that run in multiple versions of IMS must be sensitive to the version of the SXPL. The version number was SXPLVER4 in IMS 9, and SXPLVER5 in IMS 10.

The exit routines that can use the new fields are:

- Build Security Environment exit routine (DFSBSEX0)
- Early Initialization exit routine (EINIT)
- Logger exit routine (DFSFLGX0)
- IMS CQS Event exit routine (ICQSEVNT)
- IMS CQS Structure Event exit routine (ICQSSTEV)
- Non-discardable Messages exit routine (DFSNDMX0)
- OTMA Destination Resolution exit routine (DFSYPRX0)
- OTMA Input/Output Edit exit routine (DFSYIOE0)
- OTMA User Data Formatting exit routine (DFSYDRU0)
- Partner Product exit routine (DFSPPUE0)

- Resource Access Security exit routine (DFSRAS00)
- Restart exit (user-specified)
- Type 2 Automated Operator exit routine (DFSAOE00)

Exit routines that do not have exit point or interface changes and are supported in the EXITDEF parameter of the USER_EXITS section of the DFSDFxxx member can be enabled for the command functions introduced in IMS 11 by using the new version of the SXPL (SXPLVER6).

The IMS Connect exit parameter list (HWSEXPRM) is changed for IMS 11. You must reassemble and rebind the IMS Connect exit routines that use HWSEXPRM to pick up the changes.

File system paths changes

The file system paths to install IMS 11 has changed, to be more efficient during the migration process. Table 7-9 shows how the file system paths differ between IMS 10 and IMS 11.

Table 7-9 IMS 11 and IMS 10 file system paths changes

DDNAME	IMS 11 path name	IMS 10 path name
SDFSIC4J	/usr/lpp/ims/ims11/ico/IBM/	/usr/lpp/ims/ico101/IBM/
SDFSJHFS	/usr/lpp/ims/ims11/imsjava/IBM/	/usr/lpp/ims/imsjava10/IBM/
SDFSJRAR	/usr/lpp/ims/ims11/imsjava/rar/IBM/	(none)
SDFSJCIC	/usr/lpp/ims/ims11/imsjava/classic/cics/IBM/	/usr/lpp/ims/imsjava10/cics/IBM/
SDFSJCPI	/usr/lpp/ims/ims11/imsjava/classic/IBM/	(none)
SDFSJTOL	/usr/lpp/ims/ims11/imsjava/classic/dlimodel/IBM/	/usr/lpp/ims/imsjava10/dlimodel/IBM/
SDFSJSAM	/usr/lpp/ims/ims11/imsjava/ivp/IBM/	/usr/lpp/ims/imsjava10/samples/IBM/
SDFSJCPS	/usr/lpp/ims/ims11/imsjava/classic/ivp/IBM/	(none)
SDFSJXQY	(none)	/usr/lpp/ims/imsjava10/IBM/

The following file system paths, which were created and used by previous releases of IMS, are no longer used in IMS 11. You can delete these obsolete file system paths after you delete the previous release from your system.

- /usr/lpp/imsico/
- /usr/lpp/IMSICO/
- /usr/lpp/ims/imsjava91/IBM/
- /usr/lpp/ims/ico91/IBM/
- /usr/lpp/ims/imsjava91/samples/IBM/
- /usr/lpp/ims/imsjava91/cics/IBM/
- /usr/lpp/ims/imsjava91/lib/IBM/
- /usr/lpp/ims/imsjava91/dlimodel/IBM/

See the *IMS 11 Program Directory for Information Management System Transaction and Database Servers V11.0* manual for installation instructions.

Global online change migration considerations (IMS 9 only)

To use the TYPE(ACBMBR) member online change function, all IMS systems in the OLCSTAT must be IMS 10 or later.

Also, the OLCSTAT must be initialized to version 2 using the default IMS 10 or later Global Online Change utility (DFSUOLC0) or by specifying VERS=2 in the utility.

IMS restart messages (IMS 9 only)

In IMS 9, a large number of the following messages could be issued during IMS restart:

- DFS826I xxx DBD ERRORS SENT TO JOB LOG
- DFS830I BLDL FAILED FOR FOLLOWING PSBs

IMS 10 added two parameters, MSG0826 and MSG0830, to the <DIAGNOSTICS_STATISTICS> section of the DFSDFxxx PROCLIB member to indicate whether the BLDL types of DFS826I and DFS830I are issued or suppressed. The default is to issue the messages as in prior releases.

To suppress these messages, specify either or both of the following parameters:

- MSG0826=SUPPBLDL
- MSG0830=SUPPBLDL

Java class libraries for IMS migration considerations

This section discusses the general and specific migration considerations for the Java class libraries for IMS 11 and the applications that use them.

General migration considerations for the Java class libraries

The `com.ibm.ims.application.IMSApplication` class is deprecated in IMS 10 and later. Although you are not required to change your applications, doing so is good practice.

The applications that use or subclass the IMSApplication class can be modified as follows:

- Remove `extends IMSApplication` from the class declaration line. For example, if the line currently is:

  ```
  public class CustomerApplication extends IMSApplication
  ```

  The line is changed to:

  ```
  public class CustomerApplication
  ```

- The main method of the application no longer has to call the `IMSApplication.begin()` method. Instead, the main method can directly call the `public void doBegin()` method. Alternatively, move the logic from the `doBegin()` method to the main method and delete the `doBegin()` method.

New Java applications for IMS have to implement only a main method.

The `com.ibm.ims.base.DLISecondaryIndexInfo` class has been removed from the library.

This change will impact you only if you did not use the DLIModel utility plug-in to generate the metadata classes (the database view) or if you use the DLISecondaryIndexInfo class explicitly in your code.

The `com.ibm.ims.db.SecondaryIndexInfo` class has been renamed to `com.ibm.ims.base.SecondaryIndexInfo`. This change affects you only if you use the class directly in your code. The metadata that is generated by the DLIModel utility plug-in is not affected.

The `imsjava.jar` file is not supported in versions later than IMS 9. See Table 7-10 on page 278 for the JAR or RAR files that pertain to your environment.

Table 7-10 JAR and RAR files for Java applications using IMS Universal drivers

Driver	JAR or RAR file
IMS Universal DL/I driver	imsudb.jar
IMS Universal JDBC driver	imsudb.jar
IMS Universal DB resource adapter	For use within WebSphere Application Server (both z/OS and distributed platforms): ► imsudbXA.rar for CCI two-phase commit transaction support ► imsudbLocal.rar for CCI local transaction support ► imsudbJXA.rar for JCA/JDBC two-phase (XA) commit processing or local transaction processing ► imsudbJLocal.rar for JDBC local transaction processing only

Table 7-11 shows the JAR files for Java applications that use the classic Java APIs for IMS.

Table 7-11 JAR files for Java applications using classic Java APIs for IMS

Environment	JAR file
All	imsjavaBase.jar
Java message processing (JMP) region Java batch processing (JBP) region	imsjavaTM.jar
WebSphere Application Server for z/OS	imsJBJCA.jar
WebSphere Application Server for distributed platforms	imsRDSClient.jar

IMS support for XQUERY (IMS 9 only)

Although the support in IMS for XQuery takes advantage of several features of the JDK 5.0 runtime, no changes to applications are necessary to incorporate JDK 5.0.

XQuery introduces a new format for representing the XML structure of an IMS database in the defining IMS XML schema, but you are not required to migrate from the XML that was generated by IMS 9.

XML schemas that were generated by IMS 9 can be used unchanged with applications that are written for XQuery.

Java applications that access GSAM databases (IMS 9 only)

Use of the GSAM interfaces introduced in IMS 10 is optional for IMS JDBC applications. Existing JBP applications written for IMS 9 continue to run unchanged.

To use the enhancements provided in IMS 10, however, these applications have to be modified to use the following new and changed classes:

- The `com.ibm.ims.db.GSAMConnection` class provides the interface to read and write records in a GSAM database.
- The `com.ibm.ims.db.GSAMConnectionFactory` class is used to create a GSAMConnection.
- The `com.ibm.com.ims.db.GSAMRecord` class is used to represent records in a GSAM database. It provides the mapping between the data in the segment and access functions on the class.

Returning IMS database records as DB2 result sets (IMS 9 only)

Existing DB2 stored procedures that are written to accept IMS information in individual fields continue to run unchanged.

Use of the IMS 10 support for returning IMS database records as DB2 result sets for DB2 stored procedures is a good practice.

Migration considerations for Java dependent regions (IMS 9 only)

For IMS 10 and later, the following two statements are replaced with the `-Djava.class.path` statement:

```
-Dibm.jvm.shareable.applcation.class.path
-Dibm.trusted.middlware.class.path
```

Note: The Java dependent region (JDR) support that is provided by the classic Java APIs for IMS is stabilized. Migrate your applications to use the `imsutm.rar` driver (delivered with IMS 11 and higher) for this support.

IMSplex migration considerations

Migrating an IMSplex from one version of IMS to another is a complex process because of the many factors involved and the different configurations that are possible.

Although running a mixed IMSplex is possible, a good practice is to upgrade all your CSL components to IMS 11 if any control region is at IMS 11.

Control regions are limited as to which version of CQS they can connect to. The rules are listed in 7.2.5, "Common Queue Server coexistence" on page 231.

If you are running multiple LPARs, migrate one LPAR at a time.

If you are running multiple IMSs on one LPAR, migrate one IMS at a time.

IMS Connect migration considerations

In previous releases, you could specify multiple SSL ports even though only one active port at a time was supported. If a second port was opened, unpredictable results including an abend could occur. This was clearly undesirable.

In IMS 11, IMS Connect initialization fails if you specify multiple SSL ports. You must modify the HWSCFGxx member to specify only one port.

There are several new specifications in the HWS, TCPIP, and DATASTORE statements of the HWSCFGxx configuration file and these appear in the display output of commands such as VIEWHWS, VIEWPORT and VIEWDS.You should modify automation programs and MTO documentation to recognize these new fields.

Automation programs that read the output of IMS Connect displays or query the HWSP1410W message must be aware of the new information and fields that have been added by the IMS 11 enhancements. Similarly, MTOs that issue IMS Connect commands should understand that additional information is provided.

Message HWSX0908W is issued if the old exits HWSIMSO0 and HWSIMSO1 continue to be specified in the IMS Connect configuration member.

If WARNSOC and WARNINC are specified in the TCPIP HWSCFGxx statement, then new messages are issued when the warning level is reached (HWSS0772W) and when the number of sockets falls below the warning level (HWS0773I).

Because the HWSEXPRM macro has been expanded, IMS Connect exit routines that invoke the macro must be re-assembled. Remember also that the XIBDS (Exit Interface Block Data Store Entry) has also been expanded.

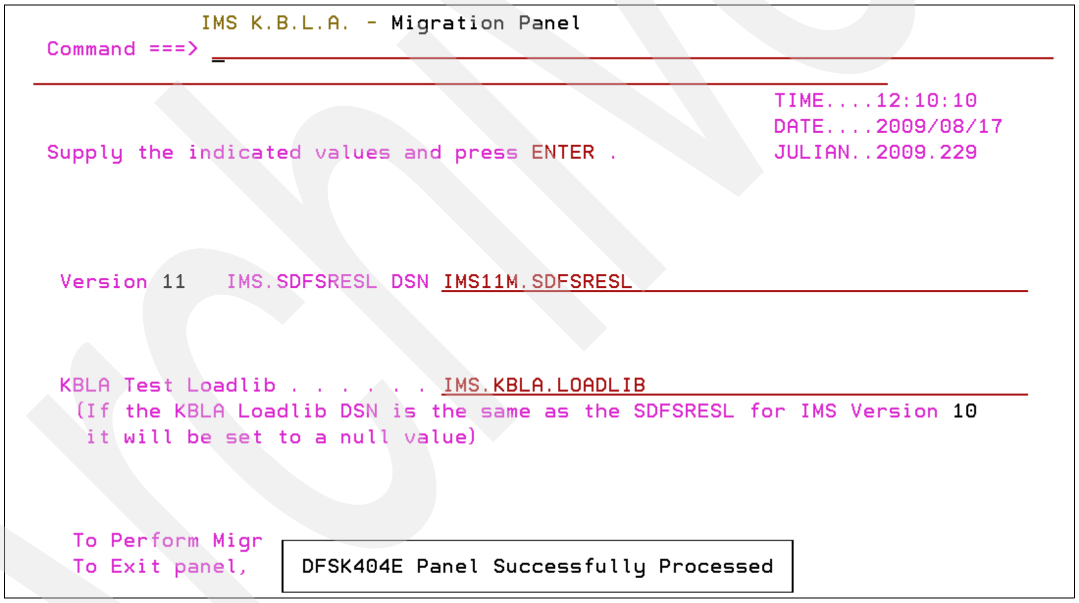
With the TCP/IP automatic reconnect capability, new code in IMS Connect's terminate port thread process automatically issue an internal OPENPORT command on a timer basis. Operator commands or automation that are issued to ensure that IMS Connect reestablishes connectivity with a TCP/IP network are no longer needed.

To take advantage of the Generated Client ID function, the IMS TM resource adapter must be replaced with the new version.

The new BPE trace capability is enabled only when the old function is disabled by removing or commenting out the HWSRCORD DD statement in the IMS Connect startup procedure. After the new function has been enabled, new RCTR entries in the BPE external trace data sets will be introduced as variable length trace entries.

Knowledge-Based Log Analysis

The first time you run a new version of Knowledge-Based Log Analysis (KBLA), the KBLA Migration panel opens, enabling you to supply a version-specific IMS.SDFSRESL data set name. Figure 7-18 shows this panel.



```
               IMS K.B.L.A. - Migration Panel
 Command ===> _
                                                          TIME....12:10:10
                                                          DATE....2009/08/17
 Supply the indicated values and press ENTER .            JULIAN..2009.229


  Version 11   IMS.SDFSRESL DSN IMS11M.SDFSRESL


  KBLA Test Loadlib . . . . . . IMS.KBLA.LOADLIB
   (If the KBLA Loadlib DSN is the same as the SDFSRESL for IMS Version 10
    it will be set to a null value)


   To Perform Migr
   To Exit panel,   DFSK404E Panel Successfully Processed
```

Figure 7-18 KBLA Migration Panel

To migrate KBLA to a new version:

1. Verify that the version-specific IMS.SDFSRESL DSN (data set name) and the KBLA Test Loadlib (test load library) names are correct, or type different values.
2. Press Enter.

The IMS KBLA Migration Panel facilitates migrating KBLA from IMS 9 or IMS 10 to IMS 11 by prompting users to enter IMS 11-related ISPF variables.

When the Migration panel displays, the IMS 11 IMS.SDFSRESL library name is initialized to `hlq.SDFSRESL`, where `hlq` is the high-level qualifier that was used to install IMS.

If you are migrating from IMS 9, the KBLA test load library displays the value from the earlier (pre-migration) version of IMS. If this KBLA test load library value is the same as the value for the currently installed IMS 9 IMS.SDFSRESL, the KBLA test load library value in the migration panel is blank.

This ensures that the IMS 11 IMS.SDFSRESL is not inadvertently overridden by the IMS 9 IMS.SDFSRESL in the JOBLIB concatenation.

If you want the IMS 11 IMS.SDFSRESL data set to override the IMS 9 IMS.SDFSRESL data set, use the ISPF KBLA Parameter Maintenance panel (option 0.1 panel) to do so.

Refer to 2.6, “KBLA” on page 50 for the enhancements in IMS 11.

Large sequential data set support enhancement (IMS 9 only)

IMS 10 introduced support for large format data sets. Migration to these data sets can be done only at data-set creation time. The following process shows one possible approach to using this support:

1. Determine which existing GSAM data sets that use the BSAM access method and which sequential OSAM data sets span more than one volume and are larger than 65535 tracks.
2. Determine if you have hardware that supports more than 65535 tracks.
3. Add the DSNTYPE=LARGE DD statement to the JCL that defines these data sets.
4. Allocate the new data sets.
5. Bring the new data sets online.

   If you are using the new data sets for an online data set (OLDS), you can bring the data set online with the /STA OLDS command.

   If you are using the new data set for a message queue, perform a cold-start of IMS to bring the new data sets online.

   If you are using the new data sets for a database rebuild it using the reorganization process (unload and reload) and use the /STA DB command or the UPD DB command to bring the new data sets online.

Program specification block (PSB) size increase (IMS 9 only)

IMS 10 and IMS 11 PSBs are 16 bytes larger *per PCB* than in IMS 9.

Because there are usually many PCBs within a PSB, this size increase is larger than it first seems. When multiple copies of a PSB are in the pool (because you have specified parallel scheduling for that PSB), the increased size is magnified.

Therefore, the larger PCBs require that you increase the space in PSB pools appropriately. You should review the settings of the PSB pool parameters CSAPSB= and DLIPSB= in DFSPBxxx.

RACF enhancements to replace SMU (IMS 9 only)

IMS 9 is the last release that supports the Security Maintenance utility (SMU) and Application Group Name (AGN) security.

To prepare for this change, IMS 9 allows you to use RACF (through the SAF interface), user exit routines, and RACF-driven resource access security (RAS) to implement security functions that previously required the use of SMU.

The IMS 9 security enhancements also eliminates the 65535 terminal limit of SMU-defined sign-on verification security. The number of terminals you can secure with RACF is not limited.

To migrate current SMU security implementations to RACF (or an equivalent product), the system programmer might have to perform some or all of the following tasks:

- Translate the current AGN definitions into RACF only definitions.
- Add the four new security classes with the default names IIMS, JIMS, LIMS, and MIMS to the installation-defined class descriptor table if the classes are not already defined as default classes in the class descriptor table that is supplied by IBM.
- Specify the new OPTIONS=SIGNON parameter on all applicable TYPE or TERMINAL system-definition macros, or specify the new SIGNON=ALL|SPECIFIC parameter in the DFSDCxxx PROCLIB member for static terminals that are currently defined through SMU as terminals that are required to sign on.
- Specify the new type-1 AOI definitions for the startup parameter AOI1= and on the system-definition TRANSACT macro for type-1 AOI and type-2 AOI.
- Change the TCO and AOI definitions to use the SAF and DFSCCMD0 exit routine for security.
- Define the new RAS parameters in the IMS system-definition SECURITY macro or the parameters on the existing parameter ISIS=.
- Change the MSC link receive security definitions to use SAF and the DFSCCMD0 exit routine for security.
- Change the use of passwords for the /LOCK and /UNLOCK commands to use SAF and the DFSCCMD0 exit routine.
- Disable resource consistency checking for the MATRIX data set if you are going to have a mixed-version IMSplex that includes IMS 9.

  To do this, specify MODBLKS in the NORSCCC= keyword of the DFSCGxxx PROCLIB member. This action disables resource consistency checking for both the MATRIX and MODBLKS data sets.

Specifying any valid value for RAS (such as RAS, RASRACF, RASEXIT, or NORAS) in the SECURITY system definition macro overrides any specifications for AGN definitions. However, AGN definitions are retained if you do not specify a RAS value (IMS uses a default of NORAS). Specifying either ISIS=N or ISIS=0 results in no RAS security and no AGN security.

The SMU to RACF Conversion utilities are a set of programs that read and convert SMU control statements and stage 1 macros to create comparable RACF control statements or system definition stage 1 macros. For more information about this topic, refer to 8.2, “Conversion utilities for SMU to RACF migration” on page 288.

You should be able to convert most of your existing SMU AGN definitions to use RACF resource group names.

RACF mixed-case password support

The default values for PSWDC and PSWDMC are different for IMS 11 than they were in earlier versions of IMS. The new default values are PSWDC=R and PSWDMC=R (use the RACF specification).

IMS 9 is not capable of processing mixed-case passwords.

IMS 10 can process mixed-case passwords, but to enable this function, you must specify PSWDC=M for IMS and PSWDMC=Y for IMS Connect. The default values for IMS 10 are PSWDC=U (uppercase) and PSWDMC=N (not mixed-case).

The PSWDC and PSWDMC parameters are enhanced in IMS 11 with the “R” specification, which means that IMS and IMS Connect should handle passwords in the same manner as is specified in RACF.

Remote Site Recovery migration

The migration of systems using Remote Site Recovery (RSR) is similar to migrations for previous releases.

IMS 11 tracking systems can process logs produced by lower releases.

The IMS 11 Isolated Log Sender (ILS) function of the Transport Manager System (TMS) can process logs created by lower releases. However, note the following information:

- IMS 10 and IMS 9 tracking systems cannot accept logs produced by IMS 11.
- IMS 10 and IMS 9 ILSs cannot accept logs produced by IMS 11.

Although you could migrate all of the RSR components at the same time, you would likely migrate them in stages. The tracking system must be migrated before or at the same time as the ILS at the active site. The ILS at the active site must be migrated before or at the same time as the active IMS system.

The RECONs must be upgraded to IMS 11 before the systems that use them are migrated to IMS 11.

Migration steps

Perform the following migration steps:

1. Upgrade the RSR tracking system RECONs to IMS 11.
2. Migrate RSR tracking system to IMS 11.
3. Upgrade the active system RECONs to IMS 11.
4. Migrate active system Transport Manager Subsystem (TMS) running Isolated Log Sender to IMS 11.
5. Migrate active IMS to IMS 11.

Serviceability enhancements

To take advantage of serviceability enhancements when you migrate to IMS 11, you might have to increase storage or modify automated tools or procedures.

Because the number of address spaces included in the system dump is limited by the amount of storage specified by the MAXSPACE parameter of the z/OS CHNGDUMP command, you might have to increase the amount of storage to accommodate the additional address spaces. However, IMS does not exceed the MAXSPACE value.

You might have to modify your automated operations tools or procedures if they are dependent on the format of the DFS064I or DFS065 messages.

Syntax Checker migration considerations

The Syntax Checker assists with IMS release-to-release migrations by providing the ability to convert supported PROCLIB members from one release to the other.

When you use the Syntax Checker to check parameters for earlier releases of IMS, you must verify that the correct release number is displayed.

Fallback

Syntax Checker is a stand-alone, offline component and it can fall back by installing the previous release of Syntax Checker.

You should always maintain a copy of your previous-version PROCLIB members to enable fallback.

Enhancements in IMS 11 are discussed in 2.4, "Syntax Checker" on page 32.

8

Security enhancements

In this chapter, we discuss benefits of the IMS Version 11 security enhancements, including:

- Mixed-case password support
- Conversion utilities for SMU to RACF migration
- Automatic signon for static terminals
- DFSDCxxx enhancement for WTOR and MTO
- DLI AUTH call enhancement

8.1 Mixed-case password support

IMS Version 11 introduces an enhancement to both IMS and IMS Connect with regard to the parameter that controls whether mixed-case password is supported. With this enhancement, both of these environments can automatically match that of RACFs by default. Mixed-case password parameters have been added to both IMS and IMS Connect, and new IMS Connect commands are also now available to manage this setting. In this section, we discuss these enhancements and how they simplify mixed-case password support implementation.

8.1.1 IMS impact

Every IMS system has a startup parameter, named PSWDC=, that indicates whether it supports mixed-case passwords. This enhancement was originally added in IMS Version 10 with two available values:

- PSWDC=U, the default, means that all passwords will be converted to upper case.
- PSWDC=M means that mixed-case passwords are allowed.

Note: IMS can use mixed-case passwords if this support is first active in the RACF environment.

New default value for PSWDC parameter

IMS Version 11 adds a third value for the PSWDC parameter, PSWDC=R. This default value means that IMS will automatically use the same value that RACF has specified for mixed-case password support. In the RACF environment, mixed-case password support is added by using the SETROPTS command. For more information about how to use this command to set mixed-case password support, refer to *IMS V10 Implementation Guide: A Technical Overview,* SG24-7526 (see information about Security enhancements in IMS Version 10). If mixed-case password support does not exist in the RACF environment, IMS converts all passwords to upper-case when PSWDC=R is set.

The benefit of this default value is that you no longer have to monitor RACF's mixed-case password value and manually keep it synchronized with IMS when it changes. Because IMS automatically uses the mixed-case password setting specified in RACF, IMS is automatically adjusted if this value changes in RACF. Managing password security is simplified because the IMS and RACF environments can now always reflect the same value, by default. Overall, IMS availability is increased because an outage is no longer required to refresh this value when it changes in RACF.

You are still able to specify PSWDC=U or PSWDC=M in the IMS environment, accepting the default of PSWDC=R is advantageous so that IMS will match RACF. Because IMS mixed-case password support functions only if the RACF environment has this same support, they should always match (to avoid errors).

8.1.2 IMS Connect impact

The change described in 8.1.1, "IMS impact" also applies to IMS Connect. The PSWDMC= parameter value defined in the IMS Connect configuration member (HWSCFGxx) indicates whether mixed-case password support is active.

Important: As a reminder, IMS Connect can have mixed-case password support if RACF is first enabled for mixed-case password support (this is the same case as with IMS).

Prior to IMS 11, the available values for this parameter were:

- PSWDMC=N, the default value, means that all passwords are converted to upper-case
- PSWDMC=Y means that mixed-case passwords are allowed.

New default value for PSWDMC parameter

IMS 11 adds a new parameter value of PSWDMC=R, which means RACF determines whether mixed-case password support is active by checking the RACF SETROPTS value. This is the default value in IMS 11; when RACF changes whether or not mixed-case password support is active, so will IMS Connect. The benefit of this enhancement is that IMS Connect and RACF can now automatically have the same value for mixed-case password support without requiring manual coordination between the two.

New parameter value for IMS Connect commands

In addition to PSWDMC=R being added to IMS Connect's HWSCFGxx configuration member, two IMS Connect commands that manage this value have also been enhanced. The IMS Connect type-1 command SETPWMC can now have a value of RCF and the z/OS Modify Interface command UPDATE MEMBER for TYPE(IMSCON) can have a parameter value of RCF for the SET(PSWDMC)() value also. The syntax diagrams for both commands is shown in Figure 8-1. In both commands, setting the value to RCF means that RACF can be checked to determine whether mixed-case password support is active.

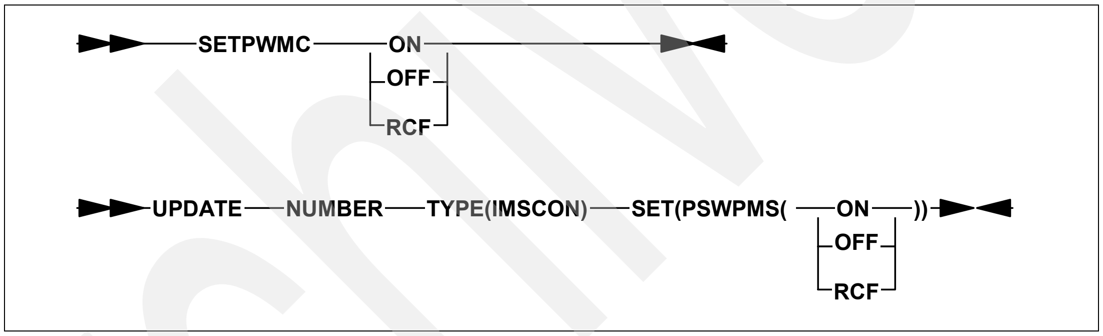


Figure 8-1 Syntax diagrams for commands that use the new RCF parameter

This enhancement is beneficial because issuing these commands with RCF specified, will result in IMS Connect and RACF automatically having the same setting for mixed-case password support. If IMS Connect specifies PSWDMC=N, and RACF changes to support mixed-case passwords, signon attempts fail until you issue the IMS Connect command with RCF specified. If you change the mixed-case password setting by using this command, note that the setting does not persist across a restart action. This happens when IMS Connect is started, and no logs are read, and therefore, the mixed-case password setting is read from the HWSCFGxx PROCLIB member.

8.1.3 Migration considerations

In IMS 11, note the default values for the IMS and IMS Connect mixed-case password support parameters. For both IMS and IMS Connect, the default parameter checks RACF to determine whether mixed-case password support is in effect. A good practice is to coordinate the mixed-case parameter settings between IMS and IMS Connect.

Regardless of whether or not mixed-case password support is active in IMS Version 10 (with a setting of PSWDC=M for IMS or PSWDMC=Y for IMS Connect), you should change the value to RACF (=R) in IMS Version 11. IMS and IMS Connect should match the RACF setting because RACF is the deciding factor for overall mixed-case password support.

8.2 Conversion utilities for SMU to RACF migration

Starting in IMS Version 10, SMU security is no longer supported. To implement IMS security, a product based on System Authorization Facility (SAF), such as RACF, must be used. IMS 11 provides conversion utilities that assist you with the migration from SMU to RACF. The utilities are also available in IMS 9 and IMS 10 through the service stream. The main purpose of the conversion utilities is to reduce the total effort required when migrating from SMU to RACF by assisting you with the steps that are time-consuming when performed manually. Specifically, the utilities assist with generating RACF resource profiles, user profiles, and modifications to Stage 1 system definition statements.

The utilities, invoked by batch processing, are also useful in that they guide you with a complete checklist of steps required for a full migration, and are especially useful if you are migrating to IMS 11 from IMS 8 or 9. These utilities read your existing SMU statements and stage 1 macros, and create equivalent RACF statements, update stage 1 macro statements and generate a migration step checklist called a *stage 1 analysis report*. Let us now examine each of these functions. You can find more information about the conversion utilities in the *IMS Version 11 System Utilities,* SC19-2446.

8.2.1 SMU to RACF syntax conversions

The utilities generate RACF statements that protect resources in RACF security classes specific to their resource type. These RACF security resource classes are listed here, in the format xIMS where x represents a resource or resource-grouping class:

- IIMS/JIMS for PSBs
- TIMS/GIMS for TRANs
- LIMS/MIMS for LTERMs

You can run the utilities to convert SMU statements to equivalent RACF statements associated with several areas of security:

- Resource access
- Logical terminal
- Automated Operator Interface (AOI)

We now discuss how the utilities work in each of these security areas.

Resource access security

In the SMU environment, PSB, TRAN, and LTERM resources are protected when they are defined in an AGN group. To protect these resources in a RACF environment, they must be protected in RACF security resource classes. We now examine how to use the conversion utilities to convert SMU `)( AGN` control statements to RACF.

Converting)(AGN statements

When using SMU security, a dependent region must have authority to access the resources within the AGN group when it is scheduled. Example 8-1 shows SMU statements prior to being converted by the conversion utility.

Example 8-1 Pre-conversion)(AGN statements

```
)(  AGN  IMSDGRP
    AGPSB   DEBS
    AGPSB   APOL1
    AGTRAN  DEBSTRN1
    AGTRAN  APOL12
    AGLTERM IMSUS02
    AGLTERM T3270LD
```

All resources that are listed are protected in the AGN group named IMSDGRP. When specifying protected resources with an AGN group, you must always specify corresponding RACF statements that grant a dependent region access to the resources contained in the group. See Example 8-2 for a sample of these types of RACF statements, which correspond to the SMU statements shown in Example 8-1.

Example 8-2 RACF statements that grant a dependent region access to AGN resources

```
ADDUSER BMPUSER1
   RDEFINE AIMS IMSDGRP OWNER(IMSADMIN) UACC(NONE)
   PERMIT IMSDGRP CLASS(AIMS) ID(BMPUSER1) ACCESS(READ)
   SETROPTS CLASSACT(AIMS)
```

In this scenario, the IMSDGRP is protected in the AIMS class. To grant the BMPUSER1 dependent region access to the resources contained within the IMSDGRP AGN group, we use RACF statements to grant the access. To convert the SMU statements to RACF statements (shown in the two preceding examples), invoke program DFSKAGN0 through JCL statements, as shown in Example 8-3.

Example 8-3 JCL that is used to convert SMU)(AGN statements

```
//RUNAGNO  EXEC  PGM=DFSKAGNO
//STEPLIB  DD DISP=SHR,DSN=IMS.SDFSRESL
//COPYFILE DD DISP=SHR,DSN=IMSBLD.IMSV91.STAGE1
//SMUFILE  DD DISP=SHR,DSN=IMSBLD.IMSV91.STAGE1(SMUMAIN)
//RACFFILE DD DISP=SHR,DSN=USERID.DFSKAGNO.RACFFILE
//USERS    DD
BMPUSER1 IMSDGRP
//CONTROL DD *
DELUSER
ADDUSER
CONNECT
//SYSPRINT DD SYSOUT=*
```

Note that the SMUFILE DD statement points to the SMU input, but that the RACFFILE DD statement points to the data set that will contain the generated RACF statements after they are written. Also note that the dependent region name BMPUSER1 is included on the following statement:

```
// USERS DD
```

After this utility has been run, a report is generated, containing statistical data about the conversion results. Example 8-4 on page 290 shows a sample report that is generated after running the `)( AGN` conversion utility.

Example 8-4 Report for)(AGN conversion utility

```
DFSKAGNO CONTROL: DELUSER
DFSKAGNO CONTROL: ADDUSER
DFSKAGNO CONTROL: CONNECT

1  DFSKAGNO: AGN SMU/RACF CONVERTER REPORT
                     DATE: 2009/086  TIME: 10:44
 NUMBER OF CONTROL RECORDS READ    :        3
 DELUSER STATEMENTS WERE GENERATED
 ADDUSER STATEMENTS WERE GENERATED
 CONNECT STATEMENTS WERE GENERATED
 NUMBER OF SMUFILE RECORDS READ    :       39
 NUMBER OF AGN GROUPS REPRESENTED  :        6
 NUMBER OF USERS                   :        6
 NUMBER OF RECORD LINES WRITTEN    :      103
 NUMBER OF NON-AGN SMU RECS IGNORED:        0
 NUMBER OF IGNORED SMU REC COMMENTS:        1
 NUMBER OF COPY STATEMENTS READ    :        2
```

Resource access security can protect resources in any of the following RACF security resource classes: TIMS, GIMS, IIMS, JIMS, LIMS, or MIMS. If you ran the utility using the JCL shown in Example 8-3 with the sample SMU)(`AGN` statements, shown in Example 8-1 as the SMU input, you would find RACF statements generated in the designated RACFFILE data set as shown in Example 8-5.

Example 8-5 RACF statements converted from)(AGN statements

```
ADDGROUP IMSDGRP OWNER (*OWNER*)
   RDEFINE JIMS USRJ0001  ADDMEM(DEBS,APOL1) UACC(NONE)
   PERMIT USRJ0001 CLASS(JIMS) ID(IMSDGRP) ACCESS(READ)
   RDEFINE GIMS USRG0001  ADDMEM(DEBSTRN1,APOL12) UACC(NONE)
   PERMIT USRG0001 CLASS(GIMS) ID(IMSDGRP) ACCESS(READ)
   RDEFINE MIMS USRM0001  ADDMEM(IMSUS02,T3270LD) UACC(NONE)
   PERMIT USRM0001 CLASS(MIMS) ID(IMSDGRP) ACCESS(READ)

ADDUSER BMPUSER1
   CONNECT BMPUSER1 GROUP(IMSDGRP)
```

After the BMPUSER1 dependent region has connected to the IMSDGRP RACF group, each time a message is scheduled in the region, RACF confirms that the region ID has authority to access the resources contained within the IMSDGRP group. As you can see, the PSBs, TRANs, and LTERMs are now protected by the separate RACF security resource-grouping classes of JIMS, GIMS, and MIMS (respectively) instead of within one all-encompassing AGN group.

Logical terminal security

In the SMU environment, logical terminal security focuses on specific LTERMs, and what types of commands and transactions are able to be issued from them. To migrate to a RACF environment, you must convert this focus from LTERM to user ID.

SMU security uses the following statements to enforce security for logical terminals, commands, and transactions:

- `)( TERMINAL`
- `)( COMMAND`
- `)( TRANSACT`

Each set of statements indicates whether an LTERM is able to issue specific commands and transactions. The conversion utilities are able to convert these SMU statements to equivalent RACF statements.

When you invoke the utilities with the JCL statements, you specify whether you want to define your LTERMs as user IDs with the TERMINAL= parameter. Specifying TERMINAL=USER can result in the generated RACF statements protecting the commands and transactions in CIMS, DIMS or TIMS and the LTERM user IDs will be authorized against these resources. If you do omit the TERMINAL= parameter from the JCL, your LTERMs will be processed as a group, and protected in the LIMS class. In this case, the commands and transactions will be treated as user IDs will be authorized to be issued from the LTERM. So, the RACF statements that are generated largely depend on how you choose to represent your LTERMs when the utilities are run. Let us now examine how to use the utilities to convert the following SMU statements to equivalent RACF statements:

- `)( TERMINAL`
- `)( COMMAND`
- `)( TRANSACT`

Invocation

To convert SMU input into equivalent RACF statements, you invoke the DFSKCIMS program using JCL statements. Many variations of RACF output are available that can be generated, depending on the parameters that you specify in the JCL. Therefore, we show just one example JCL stream for demonstration purposes and we focus on sample SMU input and RACF output in this section. Refer to Example 8-6 for the sample. To become familiar with the various types of parameters that can be invoked in the utility's JCL stream (other than what is shown here in the sample), refer to *IMS Version 11 System Utilities,* SC19-2446.

Example 8-6 JCL used to convert LTERM-based SMU)(statements into RACF statements

```
//RUNCIMS  EXEC  PGM=DFSKCIMS
//STEPLIB  DD DISP=SHR,DSN=IMS.SDFSRESL
//COPYFILE DD DISP=SHR,DSN=IMSBLD.IMSV91.STAGE1
//SMUFILE  DD DISP=SHR,DSN=IMSBLD.IMSV91.STAGE1(SMUMAIN)
//RACFFILE DD DISP=SHR,DSN=USERID.DFSKCIMS.RACFTEMP
//TIMSRDEF DD DISP=SHR,DSN=USERID.DFSKCIMS.TIMSRDEF
//CONTROL  DD *
//DIMSXREF DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*,
//  DCB=(LRECL=133,RECFM=FB,BLKSIZE=6118)
//* COMMANDS BELOW ARE INPUT TO GENERATING THE
//* RDEFINE CIMS STATEMENTS FOR EACH TYPE 1 COMMAND
//COMMANDS DD *
(ACT ALL ASS BRO CAN CHA CHE CLS COM CQC)
(CQQ CQS DBD DBR DEL DEQ DIA DIS END ERE)
(EXC EXI FOR HOL IAM IDL LOC LOG LOO MOD)
(MON MSA MSV NRE OPN PST PUR QUI RCL RCO)
(RDI REC REL RES RMC RMD RMG RMI RML RMN)
(RST RTA SEC SET SIG SMC SSR STA STO SWI)
(TES TRA UNL VUN)
```

Converting)(TERMINAL statements

Running the utility with `)( TERMINAL` SMU statements as input can protect resources in either the DIMS, CIMS or TIMS RACF security classes, depending on the TERMINAL= setting in the JCL used to invoke the utility. Refer to Example 8-7 for sample SMU input that will be converted to equivalent RACF statements after the utility has been invoked.

Example 8-7 Sample)(COMMAND SMU input statements used with the DFSKCIMS utility

```
)( TERMINAL TERM1
COMMAND START
COMMAND STOP
TRANSACT STATTRN
```

If the LTERM is processed as a user (TERMINAL=USER is specified in the JCL), the generated RACF statements are listed, as shown in Example 8-8. Note that the TERM1 LTERM is defined as a single user ID and is authorized to issue the /START and /STOP commands protected in the CIMS class and the STATTRN transaction protected in the TIMS class.

Example 8-8 RACF statements and)(TERMINAL input, LTERM is a user

```
ADDUSER TERM1 DFLTGRP(IMS) OWNER(IMS)
PERMIT STA CLASS(CIMS) ID(TERM1) ACCESS(READ)
PERMIT STO CLASS(CIMS) ID(TERM1) ACCESS(READ)
PERMIT STATTRN CLASS(TIMS) ID(TERM1) ACCESS(READ)
```

If the TERMINAL= parameter was not specified in the JCL, the TERM1 LTERM is not defined as a user, but rather as a group, and the generated RACF statements are listed, as shown in Example 8-9 and the same authorizations in Example 8-8 would occur.

Example 8-9 RACF statements and)(TERMINAL input, LTERM is not a user

```
ADDGROUP TERM1 OWNER(*OWNER* )
PERMIT STA CLASS(CIMS ) ID(TERM1 ) ACCESS(READ )
PERMIT STO CLASS(CIMS ) ID(TERM1 ) ACCESS(READ )
PERMIT STATTRN CLASS(TIMS ) ID(TERM1 ) ACCESS(READ )
```

Converting)(COMMAND statements

Running the utility with `)( COMMAND` SMU statements as input can protect resources in either the LIMS or CIMS RACF security resource classes, depending on the TERMINAL= setting in the JCL used to invoke the utility. See Example 8-10 for sample SMU input that is converted to equivalent RACF statements after the utility is invoked.

Example 8-10 Sample)(COMMAND SMU input statements used with the DFSKCIMS utility

```
)( COMMAND DISPLAY
TERMINAL TERM1
TERMINAL TERM2
TERMINAL TERM3
TERMINAL TERM4
```

If the LTERM is processed as a user (TERMINAL=USER is specified in the JCL), the generated RACF statements are listed, as shown in Example 8-11. Note that the LTERM is defined as a user and then authorized to issue the /DISPLAY command protected in the CIMS class.

Example 8-11   RACF statements and)(COMMAND input, LTERM is a user

```
ADDUSER TERM1 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT DIS CLASS(CIMS ) ID(TERM1 ) ACCESS(READ )
ADDUSER TERM2 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT DIS CLASS(CIMS ) ID(TERM2 ) ACCESS(READ )
ADDUSER TERM3 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT DIS CLASS(CIMS ) ID(TERM3 ) ACCESS(READ )
ADDUSER TERM4 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT DIS CLASS(CIMS ) ID(TERM4 ) ACCESS(READ )
```

Conversely, if the TERMINAL= parameter was not specified in the JCL, the LTERM is not defined as a user and the generated RACF statements are listed, as shown in Example 8-12. In this case, each LTERM is a protected resource in the LIMS class and the /DISPLAY command is defined as a user ID with the required first three characters of DIS. This user ID representing the /DISPLAY command is authorized against each protected LTERM in the LIMS class, authorizing each LTERM to issue it.

Example 8-12   RACF statements and)(COMMAND input, LTERM is not a user

```
RDEFINE LIMS TERM1
RDEFINE LIMS TERM2
RDEFINE LIMS TERM3
RDEFINE LIMS TERM4
ADDUSER DIS OWNER(*OWNER* ) DFLTGRP(*DEFGRP*) NOPASSWORD
PERMIT TERM1 CLASS(LIMS ) ID(DIS ) ACCESS(READ )
PERMIT TERM2 CLASS(LIMS ) ID(DIS ) ACCESS(READ )
PERMIT TERM3 CLASS(LIMS ) ID(DIS ) ACCESS(READ )
PERMIT TERM4 CLASS(LIMS ) ID(DIS ) ACCESS(READ )
```

Converting)(TRANSACT statements

Running the utility with `)( TRANSACT` SMU statements as input can protect resources in either the LIMS or TIMS RACF security resource classes, depending on the TERMINAL= setting in the JCL that is used to invoke the utility. See Example 8-13 for sample SMU input that is converted to equivalent RACF statements after the utility has been invoked.

Example 8-13   Sample)(TRANSACT SMU input statements used with the DFSKCIMS utility

```
)( TRANSACT TRAN1
TERMINAL TERM1
TERMINAL TERM2
TERMINAL TERM3
TERMINAL TERM4
```

If the LTERM is processed as a user (TERMINAL=USER is specified in the JCL), the generated RACF statements are listed, as shown in Example 8-14 on page 294. Note that the LTERM is defined as a user and then authorized to issue the TRAN1 transaction protected in the TIMS class.

Example 8-14 RACF statements and)(TRANSACT input, LTERM is a user

```
ADDUSER TERM1 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT TRAN1 CLASS(TIMS ) ID(TERM1 ) ACCESS(READ )
ADDUSER TERM2 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT TRAN1 CLASS(TIMS ) ID(TERM2 ) ACCESS(READ )
ADDUSER TERM3 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT TRAN1 CLASS(TIMS ) ID(TERM3 ) ACCESS(READ )
ADDUSER TERM4 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT TRAN1 CLASS(TIMS ) ID(TERM4 ) ACCESS(READ )
```

Conversely, if the TERMINAL= parameter was not specified in the JCL, the LTERM is not defined as a user and the generated RACF statements are listed, as shown in Example 8-15. In this case, each LTERM is a protected resource in the LIMS class and the TRAN1 transaction is defined as a user ID. The TRAN1 user ID authorized against each protected LTERM in the LIMS class, authorizing each LTERM to issue it.

Example 8-15 RACF statements and)(TRANSACT input, LTERM is not a user

```
RDEFINE LIMS TERM1
RDEFINE LIMS TERM2
RDEFINE LIMS TERM3
RDEFINE LIMS TERM4
ADDUSER TRAN1 OWNER(*OWNER* ) DFLTGRP(*DEFGRP*) NOPASSWORD
PERMIT TERM1 CLASS(LIMS ) ID(TRAN1 ) ACCESS(READ )
PERMIT TERM2 CLASS(LIMS ) ID(TRAN1 ) ACCESS(READ )
PERMIT TERM3 CLASS(LIMS ) ID(TRAN1 ) ACCESS(READ )
PERMIT TERM4 CLASS(LIMS ) ID(TRAN1 ) ACCESS(READ )
```

Static Terminal Sign-on Verification

You can also use the conversion utilities to add the SIGNON parameter to the OPTIONS keyword of your stage 1 TERMINAL macro definitions. This parameter indicates whether a static LTERM is required to signon. As a reminder, SMU security specifies) (`SIGN` SMU statements to indicate which static LTERMs transactions are required to sign on. When using RACF security, this is specified with the OPTIONS=SIGNON parameter setting on the LTERM's stage 1 TERMINAL definition. Note that as an alternative, IMS can perform signon for the LTERM automatically by using the LTERM name as the user ID. Refer to 8.3, "Automatic signon for static terminals" on page 300 for more details regarding this capability.

The conversion utilities invoke programs DFSKSMU1 and DFSKSMU2, which can read your)(`SIGN` SMU statements to determine which particular TERMINAL macro definitions require the OPTIONS=SIGNON parameter added to them. After the correct TERMINAL definitions have been found, the utilities add the OPTIONS=SIGNON parameter to the selected TERMINAL definitions if it is not there already. If OPTIONS=NOSIGNON is present, the utility does not change this setting, but instead includes a message in the summary file indicating this information. Example 8-16 shows a sample of two stage 1 TERMINAL definitions before the utilities are run.

Example 8-16 Stage 1 TERMINAL macros prior to modification by conversion utilities

```
TERMINAL NAME=TERM1,MODEL=2
   NAME LTERM1
TERMINAL NAME=TERM2,MODEL=2,
   OPTIONS=(FORCRESP,,PAGDEL),
   EDIT=(,YES),FEAT=(PFK,CARD,PEN),MSGDEL=NONIOPCB
   NAME LTERM2
```

After the utilities have been run, the stage 1 TERMINAL definitions will include the OPTIONS=SIGNON parameter, similar to Example 8-17.

Example 8-17 Stage 1 TERMINAL macros after modification by conversion utilities

```
TERMINAL NAME=TERM1,MODEL=2,OPTIONS=SIGNON
   NAME LTERM1
TERMINAL NAME=TERM2,MODEL=2,
   OPTIONS=(FORCRESP,,PAGDEL,SIGNON),
   EDIT=(,YES),FEAT=(PFK,CARD,PEN),MSGDEL=NONIOPCB
   NAME LTERM2
```

Automated Operator Interface security

When you use SMU for Automated Operator Interface (AOI) security, pairs of transactions and commands are grouped together and stored in a MATRIX data set. These valid combinations are listed in `)( CTRANS` and `)( TCOMMAND` control statements and are verified within the MATRIX before a command can be issued to IMS. Using RACF for AOI security does not use the MATRIX, but instead requires the use of an ID for security checking. The concept of pairing transactions and commands can remain the same (depending on the AOI= setting for the transaction), but with RACF security, either the transactions or commands are treated as resources and are protected in their respective classes. The ID can be a transaction, command, or user ID of the user entering the transaction. The ID requires authorization before the associated transaction can issue an AOI command (CMD) call to IMS. The resources and ID definitions will be determined by the AOI= setting for transactions defined in the TRANSACT macro.

The AOI parameter in the TRANSACT macro can be set to any of the following values:

- NO: The transaction cannot issue any commands (default setting).
- YES: The ID of the user who entered the transaction requires to be authorized to issue the command (in the CIMS/DIMS class) with a RACF PERMIT statement before it can issue the command to IMS.
- TRAN: The transaction is treated as the user ID and must be authorized to issue the protected command (in the CIMS/DIMS class) with a RACF PERMIT statement before it can issue the command to IMS.
- CMD: The command is treated as the user ID and must be authorized for action on the protected transaction (in the TIMS/GIMS class) with a RACF PERMIT statement before it can be issued to IMS. Specifically, the first three characters of the command must be defined to RACF as the user ID.

We now examine how to use the conversion utilities to convert the following SMU control statements to RACF:

- `)( CTRANS`
- `)( TCOMMAND`

Converting)(CTRANS statements

When an SMU)(`CTRANS` control statement is defined with several TCOMMAND statements under it, this indicates which commands the specified transaction can issue to IMS. See Example 8-18 for a sample of these types of SMU control statements.

Example 8-18 Pre-conversion)(CTRANS statement with several TCOMMAND statements

```
)( CTRANS AUTOCTL
    TCOMMAND START
    TCOMMAND STOP
```

In the example, the AUTOCTL transaction is able to issue the /START and /STOP commands to IMS. To convert these SMU statements to RACF statements, invoke program DFSKCIMS through JCL statements, such as those shown in Example 8-19. The form of the RACF statements is determined by the AOI= specification within the DFSKCIMS JCL, which can be set to either TRAN (the default) or CMD. The value specified must match the AOI keyword on the TRANSACT macro in stage 1 of system definition. As a reminder, the setting AOI=CMD results in the generation of RACF statements that treat the command as a user ID and the transaction as a resource. However, the setting AOI=TRAN results in the generation of RACF statements that treat the transaction as a user ID and the command as a resource.

Restriction: The DFSKCIMS program does not support the setting AOI=YES.

Example 8-19 JCL used to convert SMU)(CTRANS statements

```
//RUNCIMS  EXEC  PGM=DFSKCIMS
//STEPLIB  DD DISP=SHR,DSN=IMS.SDFSRESL
//COPYFILE DD DISP=SHR,DSN=IMSBLD.IMSV91.STAGE1
//SMUFILE  DD DISP=SHR,DSN=IMSBLD.IMSV91.STAGE1(SMUMAIN)
//RACFFILE DD DISP=SHR,DSN=USERID.DFSKAGNO.RACFFILE
//CONTROL  DD *
AOI=TRAN
//DIMSXREF DD DISP=SHR,DSN=USERID.DFSKDIMS.DIMSXRFO
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*,
//  DCB=(LRECL=133,RECFM=FB,BLKSIZE=6118)
//COMMANDS DD *
 (ACT ALL ASS BRO CAN CHA CHE CLS COM CQC)
 (CQQ CQS DBD DBR DEL DEQ DIA DIS END ERE)
 (EXC EXI FOR HOL IAM IDL INI LOC LOG LOO)
 (MOD MON MSA MSV NRE OPN PST PUR QUE QUI)
 (RCL RCO RDI REC REL RES RMC RMD RMG RMI)
 (RML RMN RST RTA SEC SET SIG SMC SSR STA)
 (STO SWI TER TES TRA UNL UPD VUN)
```

In Example 8-19, the `SMUFILE DD` statement lists the data set from which the SMU input will be read. The RACF statements that are generated are stored in the file that is specified by the `RACFFILE DD` statement. Also note that setting AOI=TRAN is included, so we know that transactions will be treated as user IDs in our new RACF statements. The `COMMANDS DD` statement simply lists the three-character representations of the IMS commands that are included in the CIMS class. After this utility has been run, a report is generated, containing statistical data about the conversion results.

Example 8-20 shows a report that is generated after running the)(`CTRANS` conversion utility.

Example 8-20 Report for DFSKCIMS conversion utility when used for)(CTRANS

```
1  DFSKCIMS: SMU COMMAND RESOURCE CONVERTER REPORT
                     DATE: 2007/092  TIME: 18:02

 AOI=CMD PROCESSING SELECTED FOR CTRANS/TCOMMAND
 NUMBER OF CONTROL RECORDS READ    :       22
 NUMBER OF SMUFILE RECORDS READ    :       18
 NUMBER OF DIMSXREF RECORDS READ   :        2
 NUMBER OF )( TERMINAL STATEMENTS  :        1
 NUMBER OF )( COMMAND STATEMENTS   :        1
 NUMBER OF )( CTRANS STATEMENTS    :        2
 NUMBER OF )( TCOMMAND STATEMENTS  :        1
 NUMBER OF COMMANDS RECORDS        :        7
 NUMBER OF TIMS ENTRIES            :        1
 NUMBER OF RECORD LINES WRITTEN    :       45
 NUMBER OF OTHER SMU RECS IGNORED  :        0
 NUMBER OF IGNORED SMU REC COMMENTS:        0
 NUMBER OF COPY STATEMENTS READ    :        0
 NUMBER OF UNSUPPORTED SMU RECS    :        0
```

If you run the utility by using the JCL shown in Example 8-19 with the example SMU)(`CTRANS` statements shown in Example 8-18 on page 296 as the SMU input, you will find RACF statements generated in the designated RACFFILE data set, as shown in Example 8-21. The /START and /STOP commands were included in the `// COMMANDS DD` statement of the JCL, which caused them to be defined to RACF.

Example 8-21 RACF statements generated from the DFSKCIMS utility with AOI=TRAN

```
ADDUSER  AUTOCTL  OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT STA      CLASS(CIMS    ) ID(AUTOCTL ) ACCESS(READ  )
PERMIT STO      CLASS(CIMS    ) ID(AUTOCTL ) ACCESS(READ  )
```

As you can see in this example, the RACF statements show the AUTOCTL transaction as a user ID, the /START and /STOP commands are listed as protected resources in the CIMS class, and the AUTOCTL transaction has been granted authorization to issue them to IMS. Alternatively, if we had specified the setting AOI=CMD in the JCL that invoked the DFSKCIMS program, we would have treated the commands as user IDs, as in Example 8-22.

Example 8-22 RACF statements generated from the DFSKCIMS utility with AOI=CMD

```
RDEFINE  TIMS     AUTOCTL  UACC(NONE  )
ADDUSER  STA     OWNER(*OWNER* ) DFLTGRP(*DEFGRP*) NOPASSWORD
PERMIT AUTOCTL  CLASS(TIMS    ) ID(STA     ) ACCESS(READ  )
ADDUSER  STO     OWNER(*OWNER* ) DFLTGRP(*DEFGRP*) NOPASSWORD
PERMIT AUTOCTL  CLASS(TIMS    ) ID(STO     ) ACCESS(READ  )
```

In this example, the /START and /STOP commands are treated as user ID and the AUTOCTL transaction is a protected resource in the TIMS class. As you can see, the commands have been granted access to be executed by the AUTOCTL transaction. The NOPASSWORD keyword has been included by default, so no password is required when the commands are issued.

Converting)(TCOMMAND statements

When an SMU)(TCOMMAND control statement is defined with several CTRANS statements under it, this indicates which transactions can issue the specified command to IMS. This approach is essentially the inverse of the discussion in "Converting)(CTRANS statements" on page 296. See Example 8-23 for a sample of these types of SMU control statements.

Example 8-23 Pre-conversion)(TCOMMAND statement with several CTRANS statements

```
)(TCOMMAND STOP
   CTRANS AUTOCTL
   CTRANS ADDINV
```

In the example, the /STOP command can be issued by the AUTOCTL and ADDINV transactions. To convert these SMU statements to RACF statements, you invoke program DFSKCIMS through JCL statements, as in Example 8-19 on page 296. As we know, the AOI= parameter specified here determines the form of the generated RACF statements.

If you ran the utility against the SMU)(TCOMMAND and CTRANS statements shown in Example 8-23, and browsed the specified RACFFILE data set, you would find the newly generated RACF statements as shown in Example 8-24 when setting AOI=TRAN is specified.

Example 8-24 RACF statements generated from the DFSKCIMS utility with AOI=TRAN

```
ADDUSER AUTOCTL OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT STO CLASS(CIMS ) ID(AUTOCTL ) ACCESS(READ )
ADDUSER ADDINV OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT STO CLASS(CIMS ) ID(ADDINV ) ACCESS(READ )
```

We see in this example that the AUTOCTL and ADDINV transactions are defined as user IDs, whereas the /STOP command has been defined as a protected resource in the CIMS class. Both transaction user IDs are authorized to issue the /STOP command with the RACF PERMIT statements.

If you specify setting AOI=CMD, the output is listed, as shown in Example 8-25.

Example 8-25 RACF statements generated from the DFSKCIMS utility with AOI=CMD

```
ADDUSER STO OWNER(*OWNER* ) DFLTGRP(*DEFGRP*) NOPASSWORD
RDEFINE TIMS AUTOCTL UACC(NONE )
PERMIT AUTOCTL CLASS(TIMS ) ID(STO ) ACCESS(READ )
RDEFINE TIMS ADDINV UACC(NONE )
PERMIT ADDINV CLASS(TIMS ) ID(STO ) ACCESS(READ  )
```

Here, we see that the /STOP command has been defined as a user ID, whereas the AUTOCTL and ADDINV transactions are protected resources in the TIMS class. The /STOP command is authorized to be issued by each transaction with the RACF PERMIT statements.

Modification to stage 1 TRANSACT macro definition

You can also use the conversion utilities to add the AOI= parameter to your stage 1 TRANSACT macro definitions. This parameter indicates whether or not a transaction is able to issue the AOI command (CMD) call to IMS. As a reminder, SMU security specifies)(CTRANS and)(TCOMMAND SMU statements to indicate which transactions can issue commands. When using RACF security, you must authorize user IDs to issue command or transaction resources, and specify the AOI= parameter on a transaction's TRANSACT definition to indicate whether the transaction, command, or signed-on user will be used as the user ID during authorization.

The conversion utilities invoke programs DFSKSMU1 and DFSKAOI1, which can read your `)(CTRANS` and `)(TCOMMAND` SMU statements to determine which particular TRANSACT macro definitions need the AOI= parameter added to them. After the correct TRANSACT definitions have been found, the utilities will add the AOI= parameter with a value that you specify in the JCL that executes the utility. Example 8-26 shows a sample of a stage 1 TRANSACT definition before the utilities are run.

Example 8-26 Stage 1 TRANSACT macro prior to modification by conversion utilities

```
TRANSACT CODE=AUTOCTL,
   PRTY=(4,4,65535),MSGTYPE=(MULTSEG,NONRESPONSE,20),
   PROCLIM=(50,5),SCHD=1,INQUIRY=(NO,RECOVER),
   FPATH=NO,MODE=SNGL,EDIT=(UC),DCLWA=YES
   APPLCTN PSB=AUTOPSB,PGMTYPE=(TP),FPATH=NO,
      SCHDTYP=SERIAL
```

After the utilities have been run, the stage 1 TRANSACT definition will include the AOI= parameter similar to Example 8-27. Note that setting AOI=TRAN is the default value, but can be set to a different value in the JCL.

Example 8-27 Stage 1 TRANSACT macro after modification by conversion utilities

```
TRANSACT CODE=AUTOCTL,
   PRTY=(4,4,65535),MSGTYPE=(MULTSEG,NONRESPONSE,20),
   PROCLIM=(50,5),SCHD=1,INQUIRY=(NO,RECOVER),
   FPATH=NO,MODE=SNGL,EDIT=(UC),DCLWA=YES,AOI=TRAN
   APPLCTN PSB=AUTOPSB,PGMTYPE=(TP),FPATH=NO,
      SCHDTYP=SERIAL
```

8.2.2 Stage 1 analysis reporting

Another function included in the conversion utilities is the ability to generate a report that contains a checklist of steps required for a full migration to RACF, some of which can be completed using other conversion utilities. Other steps must be completed manually. The Stage 1 Analysis Report utility reads stage 1 macro statements and uses this data to generate the report, which includes migration checklists for the following security areas:

- Resource access security
- AOI security
- LTERM security
- TCO security
- LOCK, UNLOCK, SET command security
- MSC link receive security
- Static terminal security

You can invoke this utility with the DFSKSTG0 program, as shown in Example 8-28.

Example 8-28 JCL to generate Stage 1 Analysis Report containing migration checklists

```
//RUNSMU2 EXEC PGM=DFSKSTGO
//STAGE1IN DD DISP=SHR,DSN=USERID.STAGE1(STGMAIN)
//COPYFILE DD DISP=SHR,DSN=USERID.STAGE1
//TEXTIN DD DISP=SHR,DSN=IMS.DFSKSAMP(DFSKSTGT)
//TEXTOUT DD DISP=SHR,DSN=USERID.DFSKSTGO.TEXTOUT
//CONTROL DD *
//SYSPRINT DD SYSOUT=*,
// DCB=(LRECL=133,RECFM=FB,BLKSIZE=6118)
```

8.3 Automatic signon for static terminals

IMS 11 includes an enhancement that enables IMS to automatically perform signon for static terminals at logon time, using the LTERM name as the user ID. This capability is enabled by specifying OPTIONS=AUTOSIGN setting on the TERMINAL or TYPE macros and has also been retrofit to IMS 10.

8.3.1 Benefits

Prior to this enhancement, RACF security was not able to be used with static terminals because of the lack of a user ID being available to use for RACF authorization. Typically, a static terminal does not have an automatic signon process with a user ID. Because IMS is now capable of automatically signing on a terminal by using the LTERM name as the user ID, it can now easily be used with RACF security for authorization to issue IMS commands. If you currently use static LTERM-based security and are migrating to RACF in preparation to move from IMS 9 to IMS 11, you can now maintain this LTERM-based setup. With the new AUTOSIGN capability, IMS creates the user IDs for you, eliminating your restriction from using RACF because of its requirement for user IDs.

Note: This enhancement has also been retrofit to IMS 10 with APAR PK85571.

8.3.2 Implementation

To use static LTERM-based security, you must specify the OPTIONS keyword parameter of AUTOSIGN on the TERMINAL or TYPE stage 1 macros that are run during system definition. Otherwise, the default OPTION of NOAUTSGN is used, which means that IMS cannot sign on the terminal at logon time. When you specify the AUTOSIGN option on the TYPE macro, the setting will apply to all subsequent TERMINAL macro statements, up to the next TYPE macro. However, when you specify the AUTOSIGN option on the TERMINAL macro, the setting applies only to that single LTERM.

Ensure that the user IDs that IMS will use in the automatic signon process (autosign) are defined to RACF. Otherwise, the following message is returned to the terminal when autosign is attempted:

```
DFS2467I <timestamp> /SIGN COMMAND REJECTED RC = 4
```

You do not have to specify a signon password to use this capability, because IMS specifies the PASSCHK=NO setting on the RACROUTE request when it performs the autosignon for the static terminal. Example 8-29 shown a TERMINAL macro definition that specifies the AUTOSIGN keyword.

Example 8-29 TERMINAL macro definition that specifies the OPTIONS=AUTOSIGN keyword

```
TERMINAL   NAME=RENDS02,TYPE=3270-A2,
            OPTIONS=AUTOSIGN,SIZE=(24,80),
            OUTBUF=256,EDIT=(YES)
NAME  RTNDSAE
```

In this example, the static terminal with node RENDS02 will be signed on at logon time with RTNDSAE as the user ID.

User exit override

You can use the logon user exit DFSLGNX0 to override the static terminal AUTOSIGN or NOAUTSGN specification that you originally set in the TERMINAL and TYPE macros. For more information about this user exit, refer to *IMS Version 11 System Utilities,* SC19-2446.

8.4 DFSDCxxx enhancement for WTOR and MTO

You are able to specify user IDs that represent the system console (WTOR) and the master terminal operator (MTO) within the DFSDCxxx PROCLIB member for transaction authorization purposes. Specify a user ID for WTOR with the WTORUSID parameter and for the MTO with the MTOUSID parameter. Although IMS does not perform signon for WTOR or the MTO, it instead creates an ACEE for your specified user IDs and uses them in the transaction authorization process.

8.5 DLI AUTH call enhancement

Prior to IMS 11, the DLI AUTH call allowed an application program to perform resource authorization checking for a user ID. When the program issued the call, IMS would perform an authorization check to determine whether the user ID was permitted to access the specified resource. In addition, if the USERDATA field was included in the DLI AUTH call, any data specific to the signed on user ID was retrieved from the RACF ACEE and returned to the application program. In IMS 11, two fields have been added to the DLI AUTH call: INSTDATA and APPLDATA, for installation-defined data and application-defined data, respectively. Both fields allow the application program to retrieve resource information and subsequently use it for display or additional security checking purposes. For further information about the DLI AUTH call, refer to *IMS Version 11 Application Programming APIs,* SC19-2429.

A

IMS 11 product and tools

This appendix discusses the following topics:

- IMS 11 packaging
- IMS Enterprise Suite
- IMS tools support for IMS 11

A.1 IMS 11 packaging

IMS 11 is product number 5635-A02 and comprises eight FMIDs, listed in Table A-1.

Table A-1 IMS 11 FMIDs

Name	FMID	Remarks
System Services	HMK1100	Required
Database Manager	JMK1101	Required for DB systems
Transaction Manager	JMK1102	Required for TM systems
Extended Terminal Option	JMK1103	Optional for TM systems
RSR Recovery-Level Tracking	JMK1104	Optional
RSR Database-Level Tracking	JMK1105	Optional for RSR Recovery level tracking.
Java on Demand	JMK1106	Optional
IRLM 2.2	HIR2200	Optional for DB systems, requires APAR PK05044

Note: IMS TM Resource Adapter (formerly known as the IMS Connector for Java) is a component of IMS Java On Demand feature FMID JMK1106.

IMS Connect is in system services component HMK1100.

For more information about installation requirements for IMS, refer to *IMS Version 11 Program Directory for Information Management System Transaction and Database Servers V11.0,* GI10-8788.

A.1.1 Hardware requirements

IMS 11 requires a z/Series machine running in z/Architecture® mode. Table A-2 shows the IBM System z processors that are supported.

Table A-2 IBM 64-bit processors

Machine name	Machine type
IBM System z9® Enterprise Class (formerly z9109)	2094
IBM System z9 Business Class (z9 BC)	2096
IBM System z10™ Enterprise Class (z10 EC)	2097
IBM System z10 Business Class (z10 BC)	2098

For more information about these processors, refer to z Hardware at:

http://www.ibm.com/systems/z/hardware/

A.1.1.1 Coupling facility requirements

Sysplex data sharing (including data caching and VSO data sharing) with IRLM V2.2 requires a coupling facility of level 9 or later.

Shared queues and shared-EMH support also require a coupling facility level 9 or later.

System-Managed duplexing requires a coupling facility level 12 or later, and bidirectional CF-to-CF links (such as HiperLink, ICB link, or IC link).

A.1.1.2 DASD requirements

The fast replication function of the Database Image Copy 2 utility (DFSUDMT0), which is new if you are migrating from IMS 9, requires DASD controllers that support either:

- The concurrent-copy feature of DFSMS
- The FlashCopy feature of the IBM Enterprise Storage Server® (ESS)
- The SnapShot feature of the IBM RAMAC Virtual Array (RVA) storage system

FlashCopy and SnapShot might require microcode from IBM to activate their functionality. Also, the source and target data sets (databases and image copies) must reside on the same ESS or RVA hardware.

Details about the DASD storage requirements are in *IMS Version 11 Program Directory for Information Management System Transaction and Database Servers V11.0,* GI10-8788.

A.1.2 Software requirements

IMS 11 requires the following software:

- z/OS 1.9 (5694-A01) or later
- High Level Assembler Toolkit 1.5 (5696-234) or later. This is a separately orderable feature of z/OS.

Depending on the features you use, IMS 11 also requires the following software:

- IMS SecureWay™ Security Server RACF feature (5694-A01) or equivalent for security. This is a separately orderable feature of z/OS.
- Rational Developer for System z 7.5 (5724-T07) for the XML adapter support in IMS Connect
- Rational Developer for System z 7.6 (5724-T07) for the support in IMS SOAP Gateway
- DFSMS Transactional VSAM (5694-A01) for Parallel RECON Access. This is a separately orderable feature of z/OS.
- TCP/IP 1.4 (5694-A01) for IMS Connect. This is a feature of z/OS.
- z/OS System SSL for Secure Sockets with IMS Connect.
- For Java dependent region (JDR) support, you must have either:
  - IBM 31-bit SDK for z/OS, Java 2 Technology Edition, V6 (5655-I98) or later
  - Sun Java SE Development Kit (JDK) V6 or later
- z/OS UNIX System Services (5694-A01) for any use of Java. This is a feature of z/OS.

A.1.2.1 Requirements for CICS

CICS requirements include:

- For database connection through DRA or transaction connection through ISC to IMS 11, you must have either:
  - CICS Transaction Server for z/OS V2.2 (5697-E93)

    If you want to use PCBLOC=31 in your DRA, you must have APAR PK54099.
  - CICS Transaction Server for z/OS V3.1 (5655-M15) or later

    If you want to use PCBLOC=31 in your DRA, you must have APAR PK54100.

- For JDBC or DL/I access from CICS using the Universal drivers, you must have either:
  - CICS Transaction Server for z/OS V2.3 (5697-E93)
  - CICS Transaction Server for z/OS V3.1 (5655-M15) or later
- For JDBC access from CICS using the classic drivers, you must have either:
  - CICS Transaction Server for z/OS V2.2 (5697-E93)
  - CICS Transaction Server for z/OS V3.1 (5655-M15) or later

A.1.2.2 Requirements for DB2

Using DB2 with IMS 11 requires:

- For External Subsystem connection to DB2, you must have:
  - DB2 UDB for z/OS 8.1 (5625-DB2) or later.
- For interoperability in a Java dependent region, you must have either:
  - DB2 UDB for z/OS 8.1 (5625-DB2) with APAR PQ74629
  - DB2 for z/OS 9.1 (5635-DB2) or later
- For Java access from a DB2 stored procedure using either the Universal or classic drivers you must have either:
  - DB2 UDB for z/OS 8.1 (5625-DB2) with APAR PQ74629 and DB2 SQLJ/JDBC driver
  - DB2 for z/OS 9.1 (5635-DB2) or later

A.1.2.3 Requirements for IMS Control Center

To use the IMS Control Center with IMS 11 you must have the DB2 Administration client, available with any of the following items:

- DB2 Universal Database™ Personal Edition 8.1 (5724-B55) or later
- DB2 Connect Personal Edition 8.1 (5724-B56) or later
- DB2 Connect Unlimited Edition 8.1 (5724-B62) or later
- DB2 Connect Application Server Edition 8.1 (5724-D54) or later
- DB2 Personal Developer's Edition 8.1 (5724-B58) or later
- DB2 Universal Database Workgroup Server Edition 8.1 (5765-F35) or later
- DB2 Universal Database Workgroup Server - Unlimited Edition 8.1 (5765-F43) or later
- DB2 Universal Database Enterprise Server Edition 8.1 (5765-F41) or later
- DB2 Connect Enterprise Edition 8.1 (5765-F30) or later
- DB2 Universal Developer's Edition 8.1 (5765-F34) or later

A.1.2.4 Requirements for Java

If you use the IMS Universal Java drivers, IMS 11 requires:

- For stand alone Java JDBC or DL/I access you must have either:
  - IBM 31-bit SDK for z/OS, Java 2 Technology Edition V5 (5655-I56) or later
  - Sun Java SE Development Kit (JDK) V5 or later
- For Java interoperability with COBOL (using either the Universal or classic drivers)
  - IBM Enterprise COBOL for z/OS and OS/390® 3.2 (5655-G53) or later

A.1.2.5 Requirements for WebSphere Application Server

To access IMS from WebSphere Application Server, you must meet these requirements:

- For access from WebSphere Application Server on z/OS using either the Universal drivers or the IMS DB Resource Adapter (classic driver) or the IMS TM Resource Adapter (also classic driver), you must have:
  - IBM WebSphere Application Server for z/OS 6.1.04 (5655-I35) or later

- For Universal JDBC driver access from WebSphere Application Server on other platforms, you must have:
  - IBM WebSphere Application Server for Distributed Platforms (5655-I35) 6.1 or later
- For IMS DB Distributed Resource Adapter (classic driver) access from WebSphere Application Server on other platforms, you must have *all of the following items*:
  - IBM WebSphere Application Server for Distributed Platforms (5655-I35) 6.1 or later
  - IBM WebSphere Application Server for z/OS 6.1.04 (5655-I35) or later
  - The z/OS Resource Recovery Services (RRS) feature, if you require two-phase commit

A.1.3 Compatibility with other versions of IMS

IMS 11 can connect or share with IMS systems at IMS 9 or IMS 10 level, if they have the correct toleration PTFs applied. This statement covers:

- Data sharing
- Communication through ISC link
- Message queue sharing
- Communication through MSC link
- Coexistence in the same IMSplex
- OTMA coexistence

A.1.3.1 Compatibility PTFs for IMS 9

Table A-3 shows the compatibility PTFs you must have if you want to run IMS 9 with IMS 11.

Table A-3 Compatibility PTFs for IMS 9

APARs	PTFs	Remarks
PK61582	UK42649	DBRC coexistence
PK66020 PK78344 PK82362 PK90462	UK42176 (PE) UK44234 UK50258 UK48851	CIMS Connect support for V9
PK53423	UK32266	Full function response mode recovery
PK27280	UK18913	When multiple versions exist in the same IMSplex
PK23402 PK32970	UK20811 UK24486	Global online change when multiple versions exist in the same IMSplex
PK30189	UK22059 (superseded by UK25099)	For QUEUE TRAN compatibility
PK47172	UK37827 (superseded by UK45124)	For OTMA shared queues coexistence (See the note following the table.)
PK40642	UK23974	DEDB randomizer XCI compatibility
PK00895	UK01650	OTMA reroute
PQ87088	UQ89762	OTMA purge undeliverable output
PK29667	UK20620	Message requeue after migration

Note: OTMA shared queues coexistence PTF is only required if all of these items are true:

- The back-end system is IMS 9.
- The application uses an alternate PCB for asynchronous output to the OTMA client.
- The OTMA client connects to the back-end (IMS 9) system to retrieve the asynchronous output.

A.1.3.2 Compatibility PTFs for IMS 10

Table A-4 shows the compatibility PTFs you must have if you want to run IMS 10 with IMS 11.

Table A-4 Compatibility PTFs for IMS 10

APARs	PTFs	Remarks
PK61583	UK42503	DBRC coexistence
PK66022 PK79204 PK82475 (open) PK87839	UK42410 (PE) UK44235 UK50263 UK48794	CIMS Connect support
PK53989	UK32360	Full function response mode recovery
PK74017 (open) PK74024 (open) PK82523 (open) PK91903 (open)	(No PTF yet) (No PTF yet) (No PTF yet) (No PTF yet)	Transaction expiration
PK70458 PK70960 PK81550	UK45082 and UK45083 UK45319 (PE) UK46298	OTMA resource monitoring
PK85571	UK49701	Static terminal auto sign-on
PK73423	UK42839	DFSMSCE0 GU entry point

A.2 IMS Enterprise Suite

IMS Enterprise Suite for z/OS, V1.1 (5655-T60) features independent components that extend IMS access and use industry standard tools and interfaces to:

- Modernize and speed application development and deployment.
- Enrich functionality.
- Ease installation and use.

The IMS Enterprise Suite components are designed to enhance your use of IMS applications and data.

In this initial release of the suite, components deliver innovative capabilities for your IMS environment that enhance connectivity, expand application development, extend standards and tools for a service-oriented architecture (SOA), ease installation, and provide simplified interfaces, as follows:

- SOAP Gateway maximizes reuse of your IMS assets, using standard interfaces with HTTP/SOAP transport to IBM and non-IBM components and use Rational tooling for XML conversion.

  SOAP Gateway enhancements use industry standards and enhanced tooling to further extend the reach and participation of your IMS assets in an SOA, with:

  – WS-Security (Web Services Security), a communications protocol for applying security to Web services, through message integrity, message confidentiality, and message authentication.

  – Business Event support, which enables WebSphere Business Events (WBE) and WebSphere Business Monitor (WBM) to receive business event data from IMS applications for business activities monitoring.

    This feature helps IMS customers leverage existing IMS assets to explore business-event processing and to explore business-event monitoring solutions, identify business problems, correct exceptions, and change processes to increase business competitiveness by improving process efficiencies.

- Connect APIs for Java and C can extend connectivity of distributed platforms to IMS and simplify application development for stand-alone user-written IMS Connect clients.

  C support is provided through the service process.

- JMS API expands Java application development in Java-dependent regions to offer synchronous callout support through the ICAL DL/I call.

- DLIModel utility plug-in provides a simplified interface for transforming IMS database information into metadata for use with application development. DLIModel utility enhancements help ease the data transformation, data integration, and tooling installation for IMS DBA, system, or application programmer by:

  – Allowing PL/I to include import in addition to COBOL copybook for users who want to integrate their existing PL/I and COBOL data structures into IMS metadata.

  – Adding PROCOPT to the DLIDatabaseView metadata class to enhance the JDBC driver performance.

  – Providing a graphical view of Virtual Foreign Key fields for JDBC programmers to understand the relational view of the data for usage of the Universal JDBC driver and Universal DB resource adapter.

  – Enhancing the usability of the current DLIModel utility wizard automatically to select DBDs referred to by a PSB, and merge existing metadata with modified PSB and DBD sources.

  – Completing the graphical editor of PSB and DBD with search, save, and print functions.

  – Seamlessly shell-sharing with other Eclipse based products from IBM with IBM Installation Manager.

- SMP/E and Installation Manager support ease of installation on z/OS and on distributed platforms.

A.2.1 Publications

These publications are available:

- *IMS Enterprise Suite for z/OS, V1.1 Program Directory*, GI10-8816
- *IMS Enterprise Suite for z/OS, V1.1 License Information*, GC19-2807

See also the Announcement letter US 209-353, available from:

http://www-306.ibm.com/common/ssi/fcgi-bin/ssialias?infotype=an&subtype=ca&appname=xldata&htmlfid=897/ENUS209-353

A.2.2 Hardware requirements

IMS Enterprise Suite for z/OS, V1.1 operates on the following hardware:

- Any 64-bit IBM processors that are capable of running the required z/OS V1.9 for functions that are to run on z/OS.
- Workstations that are capable of running Linux, AIX, or Windows XP for functions that are to run on these operating systems.
- For additional requirements (for specific items), refer to *IMS Enterprise Suite for z/OS, V1.1 Program Directory,* GI10-8788

A.2.3 Software requirements

IMS Enterprise Suite for z/OS, V1.1 component functions operate with IMS 10 (5635-A01), IMS 11 (5635-A02), or both, with the following software requirements:

- z/OS V1.9 (5694-A01) or later, for those functions running on z Servers
- IMS Enterprise Suite DLIModel utility, which runs on Windows XP, and Red Hat Linux with IMS 10 or later
- IMS Enterprise Suite JMS API open source, which runs on z/OS with IMS 10 or IMS 11 or later
- IMS Enterprise Suite SOAP Gateway, which runs on z/OS, Linux on z, AIX, or Windows:
  - Base function runs with IMS 10 or later.
  - WS-Security runs with IMS 11 or later.
  - Business Events runs with IMS 11 or later and either of the following items:
    - WebSphere Business Events Version 6.2
    - WebSphere Business Monitor Version 6.2
- IMS Enterprise Suite Connect API for Java, which runs on Windows and z/OS, with IMS 10 or later
- IMS Enterprise Suite Connect API for C, which runs on Windows, with IMS 10 or later.

For additional line item requirement information, refer to *IMS Enterprise Suite for z/OS, V1.1 Program Directory,* GI10-8788.

IMS Enterprise Suite for z/OS, V1.1 has installation-based pricing. It is a no-cost product for unlimited installations.

A.2.4 Customer requirements

IMS Enterprise Suite for z/OS, V1.1 satisfies user group requirement MR1208052022, which requested an IMS Connect API.

A.2.5 Compatibility

Note the following compatibility information:

- IMS Enterprise Suite for z/OS, V1.1 is upward compatible from IMS SOAP Gateway V10 and the IMS 10 DLIModel utility plug-in, allowing existing applications and data to be used without change.

  Migration and coexistence support is available for IMS 10 users. Review the Preventative Service Planning (PSP) information for the latest details.
- IMS 10 was the last release to include in its base the DLIModel utility.

  If you are using the IMS 10 DLIModel utility, you should migrate to the IMS Enterprise Suite for z/OS, V1.1.
- IMS SOAP Gateway V10 was the last release of that separate product.

  If you are using IMS SOAP Gateway V10, you should migrate to IMS Enterprise Suite for z/OS, V1.1.

A.3 IMS tools support for IMS 11

All current IBM IMS tools support IMS 11. In certain cases, the latest release includes IMS 11 support, in other cases you must apply a PTF for IMS 11 support.

For the most recent information, go to the following Web site and select the tab for IMS 11:

http://www.ibm.com/support/docview.wss?rs=434&uid=swg21296180

B

Configuration and workload

This appendix gives information about the configuration that we used and the sample Windows programs used for ODBM testing, which are in Appendix D, “Additional material” on page 335.

Topics in this appendix include:

- IMS configuration
- ODBM sample programs

B.1 IMS configuration

Our IMS system is an IMSPlex, composed of two control regions with shared queues and shared databases. The systems were remotely located on two LPARs systems named WTSC42 and WTSC47.

Only the control region, ODBM and IMS Connect address spaces are shown in Figure B-1, we have omitted all the other address spaces in the IMSplex for clarity. Notice that the SCI connection from IMS Connect to ODBM is by program call (PC) if they are within the same LPAR, or through XCF if they are on a different LPAR. This is the normal behavior of SCI.

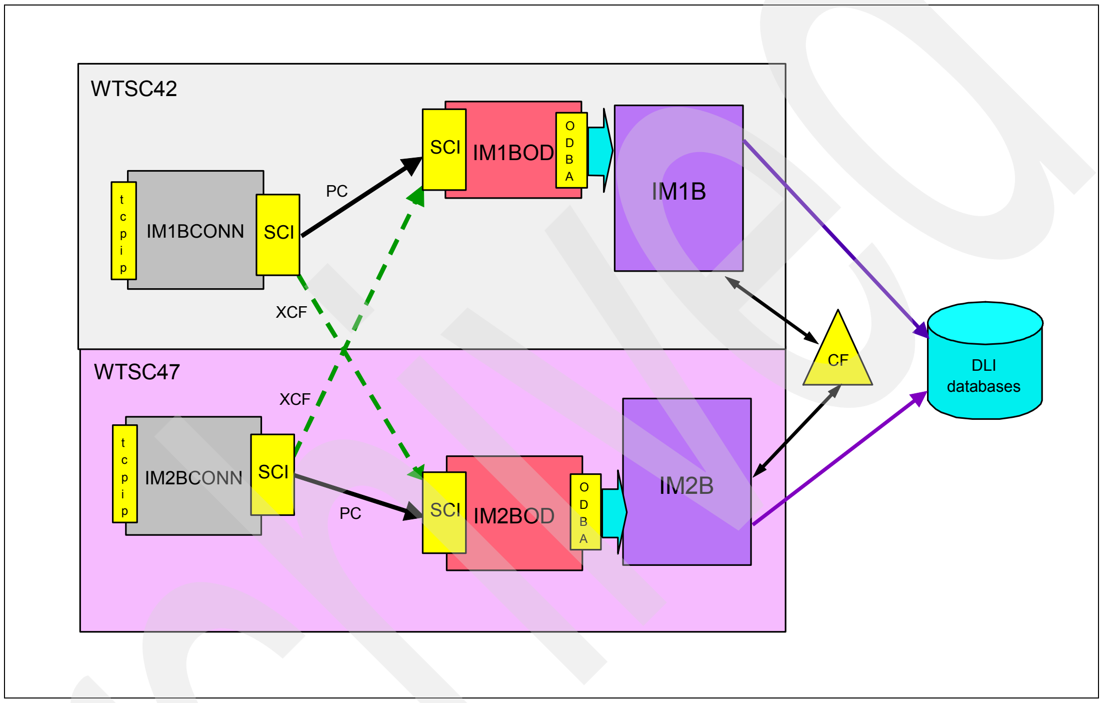


Figure B-1 The IMSplex used for this book

B.1.1 ODBM JCL and configuration members

The procedure in Example B-1 is used to start the ODBM on WTSC42. The name of ODBM is IM1BOD. The JCL for IM2BOD, which runs on WTSC47, is very similar.

Example B-1 Start procedure IMB1OD

```
//*****************************************************************
//*  ODBM  PROCEDURE FOR IM1B
//*****************************************************************
//IM1BODBM PROC RGN=3000K,SOUT=X,
//            BPECFG=BPECFGOD,
//            ODBMINIT=01B,
//            PARM1=
//*
//*-----------------------------------------------------------------*
//*     PARAMETERS:                                                 *
//*     BPECFG  - NAME OF BPE MEMBER
//*     ODBMINIT- SUFFIX FOR YOUR CSLDIxxx MEMBER
//*     PARM1   - OTHER OVERRIDE PARAMETERS FOR EXAMPLE
//*               ARMRST  - INDICATES IF ARM SHOULD BE USED
//*               ODBMNAME- NAME OF THE ODBM BEING STARTED
//*               ODBMCFG - SUFFIX FOR CONFIG MEMBER
//*               RRS     - INDICATES RRS IS (Y) OR IS NOT (N) USED
//*-----------------------------------------------------------------*
//*
//IM1BODBM EXEC PGM=BPEINI00,REGION=&RGN,
//  PARM=('BPECFG=&BPECFG,BPEINIT=CSLDINI0,ODBMINIT=&ODBMINIT',
//        '&PARM1')
//*
//STEPLIB  DD  DSN=IMSPSA.IM0B.SDFSRESL,DISP=SHR
//PROCLIB  DD  DSN=IMS11M.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
```

Example B-2 shows the initialization member, which is read by task IM1BODBM. The member for IM2BODBM is very similar.

Example B-2 ODBM initialization member CSLDI01B

```
*-------------------------------------------------------------------*
* ODBM INITIALIZATION PROCLIB MEMBER FOR IMB1ODBM                   *
*-------------------------------------------------------------------*
ARMRST=Y,                 /* SHOULD ARM RESTART ODBM ON FAILURE?    */
IMSPLEX(NAME=PLEXB)       /* NAME OF THE PLEX                       */
RRS=Y                     /* RRS NEEDED FOR ODBA (DEFAULT IS Y)     */
ODBMCFG=00B               /* IDENTIFIES MEMBER CSLDCxxx             */
ODBMNAME=IM1B             /* ODBM NAME (ODBMID = IM1BOD)            */
```

PROCLIB member CSLDC00B in Example B-3 is addressed by both ODBM tasks.

You can see from the local section that this is a configuration with two LPARs, each containing one control region. You now see this because of the two ODBM statements, but each has only one DATASTORE statement. The data store name must be the IMSID of the control region.

DRDA applications must specify one of the data store alias names to access a particular IMS. In this case, we have an alias for the whole IMSplex (IM0B) and an alias for each control region that happens to be the same as its IMSID.

Although overriding global parameters (such as the number of Fast Path buffers) is possible at an individual data store level, we have chosen not to do this.

Example B-3 PROCLIB member CSLDC00B

```
*----------------------------------------------------------------------
* ODBM CONFIGURATION MEMBER
* REFERRED TO BY ODBMCFG= PARM IN MEMBERS CSLDIxxx
*----------------------------------------------------------------------
<SECTION=GLOBAL_DATASTORE_CONFIGURATION>
TIMER=60            /* RETRY CONNECTION TO IMS EVERY 60 SECONDS     */
IDRETRY=5           /* FOR A TOTAL OF 5 TIMES                       */
*
MAXTHRDS=5          /* THREADS PER CONTROL REGION PER ODBM          */
FPBUF=6             /* FP BUFFERS PER THREAD                        */
FPBOF=5             /* FP OVERFLOW BUFFERS PER THREAD               */
CNBA=55             /* TOTAL FP BUFFERS FOR ALL THREADS             */
*----------------------------------------------------------------------
* IN THIS CONFIGURATION
* - THERE ARE TWO LPARS (AND HENCE TWO ODBM TASKS)
* - EACH LPAR CONTAINS ONLY ONE CONTROL REGION (HENCE ONLY ONE DATASTORE
*   PER ODBM)
* - THE DATASTORE(NAME=XXXX) IS THE IMSID OF THE TARGET CONTROL REGION
* - NONE OF THE GLOBAL PARAMETERS IS OVERRIDDEN AT DATASTORE LEVEL
* AN APPLICATION PROGRAM CAN SPECIFY EITHER A SPECIFIC NAME (IM1B OR
* IM2B) OR THE "GENERIC" NAME (IM0B).
*----------------------------------------------------------------------
<SECTION=LOCAL_DATASTORE_CONFIGURATION>
ODBM(NAME=IM1B,DATASTORE(NAME=IM1B,ALIAS(NAME=IM0B,NAME=IM1B)))
ODBM(NAME=IM2B,DATASTORE(NAME=IM2B,ALIAS(NAME=IM0B,NAME=IM2B)))
```

Example B-4 shows the BPE configuration statements that we used for ODBM (member BPECFGOD).

Example B-4 BPE configuration statements for ODBM

```
#
# BPE CONFIGURATION FOR ODBM ADDRESS SPACES
#
LANG=ENU
TRCLEV=(*,LOW,BPE)              /* DEFAULT BPE TRACES TO LOW */
TRCLEV=(AWE,HIGH,BPE)           /* AWE SERVER TRACE */
TRCLEV=(CBS,MEDIUM,BPE)         /* CONTROL BLKS SVCS TRACE */
TRCLEV=(DISP,HIGH,BPE,PAGES=12) /* AWE SERVER TRACE */
TRCLEV=(HASH,MEDIUM,BPE)        /* HASH TABLE TRACE */
# ODBM  TRACES
TRCLEV=(*,LOW,ODBM)
TRCLEV=(CSL,HIGH,ODBM)
TRCLEV=(ODBM,HIGH,ODBM)
```

B.1.2 IMS Connect

The procedure in Example B-5 is used to start the IMS Connect on WTSC42. We run IMS Connect under BPE, an option that became available with IMS 10.

Notice that we have configured this IMS Connect system to use *IBM IMS Connect Extensions*. Again, the JCL for IMB2CONN is very similar.

Example B-5 Start procedure IM1BCONN for IMSCON

```
//*******************************************************************
//*     IMS 11.1
//*
//* FUNCTION: START IMS CONNECT REGION WITH ODBM
//* AFFINITY WITH SC42
//*******************************************************************
//IM1BCONN PROC RGN=0M,TME=1440,SOUT=*,
//      BPECFG=BPECFGIV,HWSCFG=HWSCF01Q
//*
//HWSREGN  EXEC PGM=BPEINI00,REGION=&RGN,TIME=&TME,
//   PARM='BPECFG=&BPECFG,BPEINIT=HWSINI00,HWSCFG=&HWSCFG'
//STEPLIB  DD   DSN=CEX.V2R1M0.SCEXLINK,DISP=SHR     Connect Ext
//         DD   DSN=FUN.V2R1M0.SFUNLINK,DISP=SHR     Connect ext
//         DD   DSN=IMS11M.SDFSRESL,DISP=SHR
//PROCLIB  DD   DSN=IMS11M.PROCLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=&SOUT
//SYSUDUMP DD   SYSOUT=&SOUT
//HWSRCORD DD   SYSOUT=&SOUT
//CEXREPOS DD   DSN=CEX.V2R1M0.CEXREPOS,DISP=SHR     Connect Ext
//CEXPRINT DD   SYSOUT=&SOUT                         Connect Ext
```

Example B-6 is the IMS Connect initialization member for IM1BCONN.

It shows two ports for transaction access (7000 and 7001) and two for database access (6000 and 6001).

Two IMS control regions are defined for transaction access, IMB1 and IMB2. Transaction connection is through OTMA, so the XCF group name and the control region XCF member names are also defined. Notice also that we have defined a super member to cover both IMSs in this IMSplex.

Database connection is through the SCI to ODBM. Because this works differently, IMS Connect is not required to know the details when it starts.

The member for IM2BCONN is almost identical.

Example B-6 IMS Connect configuration member HWSCF01Q

```
HWS=(ID=IM1BHWS1,PSWDMC=N,RACF=N,RRS=Y,XIBAREA=50)
TCPIP=(EXIT=(HWSSMPL1),HOSTNAME=TCPIP,IPV6=N,PORTID=(7000,7001))
DATASTORE=(GROUP=IM0BXCF,ID=IM1B,MEMBER=IM1BHWS1,TMEMBER=SCSIM1B,
           SMEMBER=IM0B)
DATASTORE=(GROUP=IM0BXCF,ID=IM2B,MEMBER=IM2BHWS1,TMEMBER=SCSIM2B,
           SMEMBER=IM0B)
IMSPLEX=(MEMBER=IM1BHWS1,TMEMBER=PLEXB)
ODACCESS=(DRDAPORT=(ID=6000,KEEPAV=0,PORTTMOT=18000),
          DRDAPORT=(ID=6001,KEEPAV=0,PORTTMOT=18000),
          ODBMAUTOCONN=Y)
```

Example B-7 shows the BPE control statements we used with IMS Connect (member BPECFGIV).

Example B-7 BPE configuration for IMS Connect

```
*********************************************************************
* CONFIGURATION FILE FOR BPE WITH IMS CONNECT
*********************************************************************
LANG=ENU                             /* LANGUAGE FOR MESSAGES     */
                                     /* (ENU = U.S. ENGLISH)      */
#
# DEFINITIONS FOR BPE SYSTEM TRACES
#
TRCLEV=(*,HIGH,BPE,PAGES=20)         /* DEFAULT TRACES TO HIGH    */
TRCLEV=(STG,MEDIUM,BPE)              /* STORAGE TRACE             */
TRCLEV=(CBS,MEDIUM,BPE)              /* CONTROL BLK SRVCS TRACE   */
TRCLEV=(DISP,HIGH,BPE)               /* DISPATCHER TRACE          */
TRCLEV=(AWE,HIGH,BPE)                /* AWE SERVER TRACE          */
TRCLEV=(SSRV,HIGH,BPE)               /* SYSTEM SERVICE TRACE      */
#
# DEFINITIONS FOR IMS CONNECT TRACES
#
TRCLEV=(*,HIGH,HWS,PAGES=20)         /* DEFAULT TRACES TO HIGH    */
TRCLEV=(HWSI,HIGH,HWS,PAGES=100)     /* OTMA COMM ACTIVITY TRACE  */
TRCLEV=(HWSN,HIGH,HWS,PAGES=100)     /* LOCAL OPT DRIVER ACTIVITY */
TRCLEV=(HWSW,HIGH,HWS,PAGES=100)     /* TCP/IP DRIVER ACTIVITY    */
TRCLEV=(OTMA,HIGH,HWS,PAGES=100)     /* XCF CALLS TRACE           */
TRCLEV=(TCPI,HIGH,HWS,PAGES=100)     /* TCP/IP CALLS TRACE        */
```

Example B-8 shows the status of IM1BCONN after start up. It shows the ports and systems that IMS Connect has connected to. The example is the response to the following command:

```
F IM1BCONN,QUERY DATASTORE NAME (*) SHOW(ALL)
```

Example B-8 Status of IM1BCONN after start

```
HWSC0001I    DATASTORE=IM1B     STATUS=ACTIVE
HWSC0001I      GROUP=IM0BXCF  MEMBER=IM1BHWS1
HWSC0001I      TARGET MEMBER=SCSIM1B          STATE=AVAIL
HWSC0001I      DEFAULT REROUTE NAME=HWS$DEF
HWSC0001I      RACF APPL NAME=
HWSC0001I      OTMA ACEE AGING VALUE=2147483647
HWSC0001I      OTMA ACK TIMEOUT VALUE=120
HWSC0001I      OTMA MAX INPUT MESSAGE=5000
HWSC0001I      SUPER MEMBER NAME=IM0B  CMO ACK TOQ=
HWSC0001I    DATASTORE=IM2B     STATUS=ACTIVE
HWSC0001I      GROUP=IM0BXCF  MEMBER=IM2BHWS1
HWSC0001I      TARGET MEMBER=SCSIM2B          STATE=AVAIL
HWSC0001I      DEFAULT REROUTE NAME=HWS$DEF
HWSC0001I      RACF APPL NAME=
HWSC0001I      OTMA ACEE AGING VALUE=2147483647
HWSC0001I      OTMA ACK TIMEOUT VALUE=120
HWSC0001I      OTMA MAX INPUT MESSAGE=5000
HWSC0001I      SUPER MEMBER NAME=IM0B  CMO ACK TOQ=
HWSC0001I    ODBM=IM2BOD   STATUS=REGISTERED   ODBMRRS=Y
HWSC0001I      ALIAS=IM0B     STATUS=ACTIVE
HWSC0001I      ALIAS=IM2B     STATUS=ACTIVE
HWSC0001I    ODBM=IM1BOD   STATUS=REGISTERED   ODBMRRS=Y
HWSC0001I      ALIAS=IM0B     STATUS=ACTIVE
HWSC0001I      ALIAS=IM1B     STATUS=ACTIVE
```

Example B-9 shows the ODBM configuration. It is the result of the following type-2 command:

```
QRY ODBM TYPE(CONFIG) SHOW(ALL)
```

Example B-9 QRY ODBM TYPE(CONFIG) SHOW(ALL)

```
MbrName  ConfigName Global DatastoreName   CC FPBUF FPBOF CNBA MaxThrds IdRetry Timer Aliases
IM1BOD   CSLDC00B   Y                       0     6     5   55        5       5    60
IM1BOD   CSLDC00B   N      IM1B             0                                         IM0B,IM1B
IM2BOD   CSLDC00B   Y                       0     6     5   55        5       5    60
IM2BOD   CSLDC00B   N      IM2B             0                                         IM0B,IM2B
```

B.1.3 DBRC

Although we have not described DBRC in Figure B-1 on page 314, we changed the DBRC task JCL to run under BPE.

Example B-10 shows the JCL for one of the DBRCs in the IMSplex.

Example B-10 JCL to run DBRC under BPE

```
//IM1BDBRC PROC RGN=0M,SOUT=X,
//             IMSID=,
//             BPECFG=BPEDBRC,                BPE CONFIG FOR DBRC
//             DBRCINIT=00B,                  SUFFIX FOR DSPBIxxx
//             PARM1=                         OVERRIDES IF NEEDED
//*
//DBRCPROC EXEC PGM=BPEINI00,REGION=&RGN,
//             PARM=('BPECFG=&BPECFG,DBRCINIT=&DBRCINIT,IMSID=&IMSID',
//             'BPEINIT=DSPBINI0,&PARM1')
//*
//STEPLIB   DD DSN=IMSPSA.IM0B.MDALIB,DISP=SHR
//          DD DSN=IMSPSA.IM0B.SDFSRESL,DISP=SHR
//PROCLIB   DD DSN=IMSPSA.IM0B.PROCLIB,DISP=SHR
//JCLOUT    DD SYSOUT=(A,INTRDR)
//JCLPDS    DD DSN=IMSPSA.IM0B.PROCLIB,DISP=SHR
//SYSPRINT  DD SYSOUT=&SOUT
```

Example B-11 shows the DBRC parameters in member DSPBI00B.

Example B-11 DBRC configuration member

```
*---------------------------------------------------------------------*
* DBRC INITIALIZATION PROCLIB MEMBER.                                 *
*---------------------------------------------------------------------*
VSAMBUFF(INDEX=60,DATA=120)            /* VSAM buffers for RECON     */
```

Example B-12 shows the BPE control statements we used for DBRC in member BPEDBRC.

Example B-12 BPE configuration member for DBRC

```
#
# BPE configuation member for DBRC (new with IMS 11)
#                                           Andrew Wilkinson 29.07.2009
#
LANG=ENU                              /* U.S. English                */
#
# External trace datasets - one per address space
```

```
#
EXTTRACE (GDGDEF (DSN(IMS11M.BPETRACE.&JOBNAME.) BLKSIZE(32760)
                      SPACE(15) SPACEUNIT(TRK))
                  IOBUFS(10))
#
# Statistics user exit called every 600s
#
STATINTV=600
#
# Definitions for BPE system traces
#
TRCLEV=(AWE,HIGH,BPE)                /* AWE server trace on high    */
TRCLEV=(CBS,MEDIUM,BPE)              /* Ctrl blk serv trc on medium */
TRCLEV=(DISP,HIGH,BPE)               /* Dispatcher trace on high    */
#
# Definitions for DRBC traces
#
TRCLEV=(ERR,HIGH,DBRC)               /* DBRC error trace            */
TRCLEV=(RQST,LOW,DBRC,PAGES=10)      /* DBRC request trace          */
TRCLEV=(MODF,LOW,DBRC)               /* DBRC module flow trace      */
TRCLEV=(GRPS,LOW,DBRC)               /* DBRC group services trace   */
#
# User exit list PROCLIB member specifications
#
EXITMBR=(BPEDBRCE,BPE)               /* BPE  user exit definitions  */
```

Example B-13 shows the BPE exit definitions for DBRC in member BPEDBRCE.

Example B-13 BPE exits definitions for DBRC

```
#
# Sample BPE user exit list
#
# This is where we would define a BPE init/term exit routine:
#
#EXITDEF(TYPE=INITTERM,EXITS=(MYINIT00))
#
# Define BPE Statistics exit routine(s):
#
EXITDEF(TYPE=STATS,EXITS=(MYDSTAT0))
```

B.1.4 Structured Call Interface

The SCI is not new with IMS 11; we include its JCL and configuration member here because you might not already have configured a CSL.

Example B-14 on page 321 shows the JCL we used for one of our SCIs. The other SCI is very similar.

An example is also available in the SDFSISRC library, member CSLSCI.

Example B-14 SCI task JCL

```
//*------------------------------------------------------------------*
//*  SCI   PROCEDURE FOR IM1B                                        *
//*------------------------------------------------------------------*
//IM1BSC PROC RGN=3000K,SOUT=X,
//             BPECFG=BPECFGSC,
//             SCIINIT=01B,
//             PARM1=
//*
//*------------------------------------------------------------------*
//*     PARAMETERS:                                                  *
//*     BPECFG  - NAME OF BPE MEMBER                                 *
//*     SCIINIT - SUFFIX FOR YOUR CSLSIxxx MEMBER                    *
//*------------------------------------------------------------------*
//*
//IM1BSC   EXEC PGM=BPEINI00,REGION=&RGN,
//  PARM=('BPECFG=&BPECFG,BPEINIT=CSLSINI0,SCIINIT=&SCIINIT,&PARM1')
//STEPLIB  DD  DSN=IMSPSA.IM0B.SDFSRESL,DISP=SHR
//PROCLIB  DD  DSN=IMSPSA.IM0B.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
```

Example B-15 shows the SCI configuration statements that we used (member CSLSI01B).

Another example is in member CSLSI000 of the SDFSISRC library. A full discussion on this topic is in the section about CSLSIxxx in *IMS Version 11 System Definition,* GC19-2444.

Example B-15 SCI configuration

```
*-------------------------------------------------------------------
* SCI Initialization PROCLIB member
*-------------------------------------------------------------------
ARMRST=Y,                    /* SHOULD ARM RESTART SCI ON FAILURE? */
SCINAME=IM1B,                /* SCI NAME (SCIID = xxxxSC)          */
IMSPLEX(NAME=PLEXB)          /* IMSPLEX NAME (CSLxxxxx)            */
```

Example B-16 shows the BPE configuration statements that we used for SCI (member BPECFGSC).

Example B-16 BPE configuration statements for SCI

```
LANG=ENU
TRCLEV=(*,LOW,BPE)                /* DEFAULT BPE TRACES TO LOW */
TRCLEV=(AWE,HIGH,BPE)             /* AWE SERVER TRACE */
TRCLEV=(CBS,MEDIUM,BPE)           /* CONTROL BLKS SVCS TRACE */
TRCLEV=(DISP,HIGH,BPE,PAGES=12)   /* AWE SERVER TRACE */
TRCLEV=(HASH,MEDIUM,BPE)          /* HASH TABLE TRACE */
# SCI  TRACES
TRCLEV=(*,LOW,SCI)
TRCLEV=(CSL,HIGH,SCI)
TRCLEV=(SCI,HIGH,SCI)
TRCLEV=(PLEX,HIGH,SCI)
TRCLEV=(INTF,HIGH,SCI)
```

B.1.5 Operations Manager

The OM is not new with IMS 11 and not strictly necessary for ODBM. But you might have difficulty operating ODBM without an Operations Manager, so we have included it for completeness.

Example B-17 shows the JCL we used to start one of our OMs. The other OM is very similar.

An example is also available in SDFSISRC library, member CSLOM.

Example B-17 Operations Manager task JCL

```
//*****************************************************************
//*  OM  PROCEDURE
//*
//*     PARAMETERS:
//*     BPECFG  - NAME OF BPE MEMBER
//*     OMINIT  - SUFFIX FOR YOUR CSLOIxxx MEMBER
//*     PARM1   - OTHER OVERRIDE PARAMETERS
//*
//*****************************************************************
//IM1BOM PROC RGN=6000K,SOUT=X,
//            RESLIB='IMSPSA.IMOB.SDFSRESL',
//            BPECFG=BPECFGOM,
//            OMINIT=01B,
//            PARM1=
//*
//IM1BOM  EXEC PGM=BPEINI00,REGION=&RGN,
// PARM=('BPECFG=&BPECFG,BPEINIT=CSLOINI0,OMINIT=&OMINIT,&PARM1')
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//PROCLIB  DD  DSN=IMSPSA.IMOB.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
```

Example B-18 shows the OM control statements we used (member CSLOI01B).

Notice that you must install SDFSDATA on any system where the OM will run. You can also see that we are not using the OM audit log (the AUDITLOG= keyword is missing). If you want more information about OM command security, refer to the information about CSL OM command security in *IMS Version 11 System Administration,* SC19-2443.

Another example is in member CSLOI000 of the SDFSISRC library and a full discussion is in the section about CSLOIxxx, in *IMS Version 11 System Definition,* GC19-2444.

Example B-18 Operations Manager configuration

```
*-----------------------------------------------------------------
* OM Initialization PROCLIB member
*-----------------------------------------------------------------
ARMRST=Y,                    /* SHOULD ARM RESTART OM ON FAILURE?  */
CMDLANG=ENU,                 /* USE ENGLISH FOR COMMMAND DESC      */
CMDSEC=R,                    /* COMMAND SECURITY VIA CLASS OPERCMDS*/
OMNAME=IM1B,                 /* OM NAME (OMID = xxxxOM)            */
IMSPLEX(NAME=PLEXB)          /* IMSPLEX NAME (CSLxxxxx)            */
CMDTEXTDSN=IMSPSA.IMOB.SDFSDATA /* COMMAND TEXT DATASET            */
```

Example B-19 shows the BPE configuration we used for OM (member BPECFGOM).

In fact, consolidating your BPE statements into a single member for all BPE address spaces is possible. Refer to information about BPE configuration parameter member of the IMS PROCLIB data set, which is in *IMS Version 11 System Definition,* GC19-2444.

Example B-19 BPE configuration statements for OM

```
LANG=ENU
TRCLEV=(*,LOW,BPE)              /* DEFAULT BPE TRACES TO LOW */
TRCLEV=(AWE,HIGH,BPE)           /* AWE SERVER TRACE */
TRCLEV=(CBS,MEDIUM,BPE)         /* CONTROL BLKS SVCS TRACE */
TRCLEV=(DISP,HIGH,BPE,PAGES=12) /* AWE SERVER TRACE */
TRCLEV=(LATC,LOW,BPE)           /* LATCH TRACE  */
TRCLEV=(HASH,MEDIUM,BPE)        /* LATCH TRACE  */
# OM  TRACES
TRCLEV=(*,LOW,OM)
TRCLEV=(CSL,HIGH,OM)
TRCLEV=(OM,HIGH,OM)
TRCLEV=(PLEX,HIGH,OM)
```

B.2 ODBM sample programs

Several samples are available to download. See “Additional material” on page 335 for instructions on how to download these samples.

The `OpenDBSamples.zip` file contains five programs, two of them include files and show various Java programming techniques.

- The program code in `OpenDBIVP.java` file resides in a Rational Application Developer project and was used to test aspects of ODBM. It uses a ConnectionFactory and DLI style access.
- The `OpenDBIVPCCIDLI.java` file is a program using IMSManagedConnectionFactory and DLIInteractionSpec access.
- The `OpenDBIVPCCISQL.java` file uses the IMSManagedConnectionFactory with SQLInteractionSpec access.
- The `OpenDBJDBCSample1.java` file and `OpenDBJDBCSample2.java` file use JDBC access, using the type-4 IMS Universal driver.

All the programs are standalone with a main entry point, and must run as a Java Application.

The remaining two files (`DFSIVPDBView.java` and `DFSSAM09DBView.java`) contain class definitions for accessing the existing IMS sample databases DI21PART and IVPDB1.

C

z/OS samples

This appendix describes the downloadable z/OS sample programs mentioned elsewhere in this book. Instructions for how to download these items are in Appendix D, “Additional material” on page 335.

Topics in this appendix include:

- ► Sample DRDA program
- ► Sample ODBM program
- ► Sample BPE DBRC statistics exit

C.1 Sample DRDA program

File `DRDA1.EXEC.TXT` is a simple REXX program which uses TCP/IP to send DRDA requests to IMS Connect. It retrieves segments from a sample database. DRDA1 runs on z/OS, which means it need not worry about ASCII to EBCDIC conversion.

This program is not intended as an example of good programming, merely to illustrate one way of using DRDA with IMS.

The program:

1. Exchanges information with IMS Connect (EXCSAT chained with ACCSEC).
2. Passes a user ID and password to IMS Connect for verification (SECCHK).
3. Allocates the PSB DFSSAM02 (ACCRDB).

   The program does not specify an alias name, instead it allows IMS Connect to select an ODBM for the database access.
4. Retrieves root segments from DI21PART whose part numbers are in the range indicated by records in file SYSIN (OPQNRY chained with INAIB, DLIFUNC and SSALIST).

   The program retrieves 12 roots at a time.
5. Commits (RDBCMM).
6. Deallocates the PSB (DEALLOCDB).

If you are already familiar with DRDA programming, you will notice that programming for IMS is different in many ways. Extensive information about the IMS use of DRDA is in the DRDA DDM command architecture area of *IMS Version 11 Application Programming APIs,* SC19-2429.

You should read that information in conjunction with the *DRDA, Version 4, Volume 3: Distributed Data Management (DDM) Architecture*, available from The Open Group at:

http://www.opengroup.org

Another way to learn about IMS DRDA flows is to see what the IMS Universal drivers do. You can use the IMS Connect Recorder Trace to see the DRDA flow from the ODBM IVP program (which is in Java and runs on z/OS).

The JCL to execute program DRDA1 is in Example C-1 on page 327, where:

- The `user-exec-library` is the library where you stored DRDA1
- The `ip-address` is the IP address of IMS Connect
- The `port-number` is one of IMS Connect's DRDA ports (defined on the ODACCESS statement in the IMS Connect configuration member).
- The `password` is the password of the user submitting the job (this is needed if you perform verification in IMS Connect).

You can specify different part numbers in file SYSIN.

Example C-1 JCL to execute sample program DRDA1

```
//DRDA1    EXEC PGM=IKJEFT01,PARM=DRDA1
//SYSEXEC   DD DISP=SHR,DSN=user-exec-library
//SYSTSPRT  DD SYSOUT=*
//SYSTSIN   DD *
ADDRESS = 'ip-address'; PORT = 'port-number';
PW = 'password';
//SYSIN     DD *
AN960C10 CSR13G104KL
NOT_THERE ZZZZZ
803008035 9999
```

If you run the program, you will get a report like Example C-2. We have not printed the whole report, just enough for you to see how the program works. Notice that some of the longer lines have wrapped. The actual report is prettier.

Example C-2 Sample output from program DRDA1

```
Sending: EXCSAT
0031 D051 0001 002B 1041 - 000B 115E ANDREWW
                         - 0015 115A EXEC DRDA1 v1.1.0
                         - 0007 1147 DFS
0010 D001 0001 000A 106D - 0006 11A2 0003
Response received: Length = 73
0039 D043 0001 0033 1443 - 0011 116D IMOBHWS1.
                         - 000B 115E ANDREWW
                         - 000C 115A 5635-A02
                         - 0007 1147 DFS
0010 D003 0001 000A 14AC - 0006 11A2 0003
--------------------------------------------------------------------------------
Sending: SECCHK
0027 D001 0001 0021 106E - 0006 11A2 0003
                         - 000B 11A0 ANDREWW
                         - 000C 11A1 ********
Response received: Length = 21
0015 D002 0001 000F 1219 - 0006 1149 0000
                         - 0005 11A4 00
--------------------------------------------------------------------------------
Sending: ACCRDB
003E D001 0001 0038 2001 - 000C 2110 DFSSAM02
                         - 0006 210F 2407
                         - 0015 112E EXEC DRDA1 v1.1.0
                         - 000D 002F QTDSQL370
Response received: Length = 57
0039 D002 0001 0033 2201 - 0008 112E
                         - 0006 1149 0000
                         - 0014 2135 25A93198 25AB5050 14A09345 46656E40
                         - 000D 002F QTDSQL370
--------------------------------------------------------------------------------
0026 D051 0001 0020 200C - 0008 2156 0000000C
                         - 000C C907 DBPCB01
                         - 0008 2114 00001000
001E D053 0001 0018 CC01 - 000C C901 DBPCB01
                         - 0008 C904 00000100
0012 D053 0001 000C CC05 - RETRIEVE
0055 D003 0001 004F CC06 - 0006 C905 0001
                         - 0045 C906 PARTROOT(PARTKEY >=02AN960C10       &PARTKEY
<=02CSR13G104KL    )
Response received: Length = 447
```

```
0010 D052 0001 000A 2205 - 0006 1149 0000
001C D053 0001 0016 241A - 0676D026 00000671 E0D00001 0671F0E0 0000
0157 D053 0001 0151 241B - aiboalen 00000032 aibretrn 00000000 aibreasn 00000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '
seg name PARTROOT
                         - key feedback '02AN960C10       '
            Segment data - '02AN960C10                 WASHER                 '
                         - aiboalen 00000032 aibretrn 00000000 aibreasn 00000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '
seg name PARTROOT
                         - key feedback '02CK05CW181K     '
            Segment data - '02CK05CW181K               CAPACITOR              '
                         - aiboalen 00000032 aibretrn 00000000 aibreasn 00000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '
seg name PARTROOT
                         - key feedback '02CSR13G104KL    '
            Segment data - '02CSR13G104KL              KR1J50KS               '
003C D002 0001 0036 220B - 0006 1149 0004
                         - 002C CC02 aiboalen 00000000 aibretrn 00000900 aibreasn 00000000
aiberrxt 00000000
                                     db name  DI21PART seg lev  00       status 'GE'
seg name
--------------------------------------------------------------------------------
Sending: OPNQRY
0026 D051 0001 0020 200C - 0008 2156 0000000C
                         - 000C C907 DBPCB01
                         - 0008 2114 00001000
001E D053 0001 0018 CC01 - 000C C901 DBPCB01
                         - 0008 C904 00000100
0012 D053 0001 000C CC05 - RETRIEVE
0055 D003 0001 004F CC06 - 0006 C905 0001
                    - 0045 C906 PARTROOT(PARTKEY >=02NOT_THERE     &PARTKEY <=02ZZZZZ
)
Response received: Length = 336
0010 D052 0001 000A 2205 - 0006 1149 0000
001C D053 0001 0016 241A - 0676D026 00000671 E0D00001 0671F0E0 0000
00E8 D053 0001 00E2 241B - aiboalen 00000032 aibretrn 00000000 aibreasn 00000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '
seg name PARTROOT
                         - key feedback '02N51P3003F000   '
            Segment data - '02N51P3003F000             SCREW                  '
...
            Segment data - '02RC07GF273J               RESISTOR               '
003C D002 0001 0036 220B - 0006 1149 0004
                         - 002C CC02 aiboalen 00000000 aibretrn 00000900 aibreasn 00000000
aiberrxt 00000000
                                     db name  DI21PART seg lev  00       status 'GE'
seg name
--------------------------------------------------------------------------------
Sending: OPNQRY
0026 D051 0001 0020 200C - 0008 2156 0000000C
                         - 000C C907 DBPCB01
                         - 0008 2114 00001000
001E D053 0001 0018 CC01 - 000C C901 DBPCB01
                         - 0008 C904 00000100
0012 D053 0001 000C CC05 - RETRIEVE
```

```
0055 D003 0001 004F CC06 - 0006 C905 0001
                    - 0045 C906 PARTROOT(PARTKEY >=02803008035     &PARTKEY <=029999
)
Response received: Length = 1386
0010 D052 0001 000A 2205 - 0006 1149 0000
001C D053 0001 0016 241A - 0676D026 00000671 E0D00001 0671F0E0 0000
053E D003 0001 0538 241B - aiboalen 00000032 aibretrn 00000000 aibreasn 000000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '
seg name PARTROOT
                         - key feedback '02803008035      '
            Segment data - '02803008035              GASKET                 ' ...
            Segment data - '0282124-056              RN60C3161F             '
...
            Segment data - '0282124-640              RN65C9092F             '
...
            Segment data - '0282125-869              RN75C8252F             '
...
            Segment data - '0284353-456              RW67V472               '
...
            Segment data - '0290-3033334             BONDED ASSY            '
...
            Segment data - '0290-3033665             BONDED ASSY            '
...
            Segment data - '02905537-384             CAPACITOR              '
...
            Segment data - '02906028-040             CAPACITOR              '
...
            Segment data - '02907021-782             CAPACITOR              '
...
            Segment data - '02922294-002             CONNECTOR              '
...
                         - aiboalen 00000032 aibretrn 00000000 aibreasn 00000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '     seg name
PARTROOT
                         - key feedback '02922399-001     '
            Segment data - '02922399-001             CONNECTOR              '
--------------------------------------------------------------------------------
Sending: OPNQRY (subsequent)
0026 D051 0001 0020 200C - 0008 2156 0000000C
                         - 000C C907 DBPCB01
                         - 0008 2114 00001000
001E D053 0001 0018 CC01 - 000C C901 DBPCB01
                         - 0008 C904 00000100
0012 D053 0001 000C CC05 - RETRIEVE
0055 D003 0001 004F CC06 - 0006 C905 0001
                    - 0045 C906 PARTROOT(PARTKEY  >02922399-001    &PARTKEY <=029999
)
Response received: Length = 1386
0010 D052 0001 000A 2205 - 0006 1149 0000
001C D053 0001 0016 241A - 0676D026 00000671 E0D00001 0671F0E0 0000
053E D003 0001 0538 241B - aiboalen 00000032 aibretrn 00000000 aibreasn 00000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '
seg name PARTROOT
                         - key feedback '02925363-136     '
            Segment data - '02925363-136             DIODE ZENER            '
...
            Segment data - '02925380-101             DIODE                  '
```

```
...
            Segment data - '02930331-102              FILTER                  '
...
            Segment data - '02930331-123              FILTER                  '
...
            Segment data - '02930333-001              DISCRIMINATO            '
...
            Segment data - '02946325-086              PIN                     '
...
            Segment data - '02950060-006              RELAY                   '
...
            Segment data - '02954017-001              RESISTOR                '
...
            Segment data - '02958007-180              RESISTOR                '
...
            Segment data - '02960528-067              RESISTOR                '
...
            Segment data - '02968534-001              SOCKET                  '
                         - aiboalen 00000032 aibretrn 00000000 aibreasn 00000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '     seg name
PARTROOT
                         - key feedback '02974810-010     '
            Segment data - '02974810-010              THERMOSTAT              '
--------------------------------------------------------------------------------
Sending: OPNQRY (subsequent)
0026 D051 0001 0020 200C - 0008 2156 0000000C
                         - 000C C907 DBPCB01
                         - 0008 2114 00001000
001E D053 0001 0018 CC01 - 000C C901 DBPCB01
                         - 0008 C904 00000100
0012 D053 0001 000C CC05 - RETRIEVE
0055 D003 0001 004F CC06 - 0006 C905 0001
                    - 0045 C906 PARTROOT(PARTKEY  >02974810-010    &PARTKEY <=029999
)
Response received: Length = 336
0010 D052 0001 000A 2205 - 0006 1149 0000
001C D053 0001 0016 241A - 0676D026 00000671 E0D00001 0671F0E0 0000
00E8 D053 0001 00E2 241B - aiboalen 00000032 aibretrn 00000000 aibreasn 00000000 aiberrxt
00000000
                         - db name  DI21PART seg lev  01       status   '  '     seg name
PARTROOT
                         - key feedback '02975105-001     '
            Segment data - '02975105-001              TRANSFORMER             '
...
            Segment data - '02975105-001              TRANSFORMER             '
...
            Segment data - '02989036-001              TRANSFORMER             '
003C D002 0001 0036 220B - 0006 1149 0004
                         - 002C CC02 aiboalen 00000000 aibretrn 00000900 aibreasn 00000000
aiberrxt 00000000
                                     db name  DI21PART seg lev  00       status   'GE'
seg name
--------------------------------------------------------------------------------
Sending: RDBCMM
000A D001 0001 0004 200E
Response received: Length = 43
002B D002 0001 0025 220C - 0005 2115 01
                         - 0006 1149 0004
                         - 0016 CC02 aiboalen 00000000 aibretrn 00000000 aibreasn 00000000
aiberrxt 00000000
--------------------------------------------------------------------------------
Sending: DEALLOCDB
0016 D001 0001 0010 C801 - 000C 2110 DFSSAM02
Response received: Length = 50
0032 D002 0001 002C CA01 - 000C 2110 DFSSAM02
                         - 0006 1149 0000
                         - 0016 CC02 aiboalen 00000000 aibretrn 00000000 aibreasn 00000000
aiberrxt 00000000
```

C.2 Sample ODBM program

File ODBM1.ASM.TXT is a simple assembler program which uses the ODBM API to retrieve segments from a sample database. It uses many IMS macros and so you will need the SDFSMAC library to assemble it.

This program is not intended as an example of good programming, merely to help you get started with the ODBM CSLDMI API.

The program:

1. Registers with the SCI.
2. Informs the SCI that it is ready to accept messages.
3. Registers with any available ODBM.
4. Allocates the PSB DFSSAM02.
5. Reads segments from DI21PART whose part numbers are in file SYSIN.
6. Commits.
7. Releases the PCB buffer acquired by the DLI calls.
8. Deallocates the PSB.
9. Releases the buffer acquired by the registration with ODBM.
10. Deregisters with ODBM.
11. Informs the SCI that it is no longer accepting messages.
12. Deregisters with the SCI.

Note: This program does not synchronize updates between different LPARs (because it makes no updates).

To synchronize updates between different LPARs, you must establish an RRS URTOKEN This requires APF authorization and running in supervisor state, which is beyond the scope of an application program.

A full description of the CSLDMI interface that this program uses is in section "Writing a CSL ODBM client" of *IMS Version 11 System Programming APIs,* SC19-2445.

The JCL to execute this program is shown in Example C-3 on page 332, where:

- The `plex-name` is the 5 character name of the IMSplex you want to connect to.
- The `user-load-library` is the library where you stored ODBM1.

You can specify different part numbers in the `SYSIN` file.

Example C-3 JCL to execute sample program ODBM1

```
//ODBM1    EXEC PGM=ODBM1,PARM=plex-name
//STEPLIB   DD DISP=SHR,DSN=IMS.SDFSRESL
//          DD DISP=SHR,DSN=user-load-library
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
AN960C10
NOT_THERE
250236-001
```

If you run the program, a report is issued, similar to Example C-4.

Example C-4 Sample output from program ODBM1

```
SCI REGISTER call. Return code=00000000, reason=00000000
SCI READY    call. Return code=00000000, reason=00000000
ODBM REG     call. Return code=00000000, reason=00000000
Connected to IM2BOD
ODBM APSB    call. Return code=00000000, reason=00000000
                   AIB return code=0000, reason=0000, extension=00000000
ODBM GU      call. Return code=00000000, reason=00000000
                   AIB return code=0000, reason=0000, extension=00000000
Segment retrieved=02AN960C10               WASHER
ODBM GU      call. Return code=00000000, reason=00000000
                   AIB return code=0900, DLI status='GE'
ODBM GU      call. Return code=00000000, reason=00000000
                   AIB return code=0000, reason=0000, extension=00000000
Segment retrieved=02250236-001             CAPACITOR
ODBM COMMIT  call. Return code=00000000, reason=00000000
PCB RELEASE  call. Return code=00000000, reason=00000000
ODBM DPSB    call. Return code=00000000, reason=00000000
                   AIB return code=0000, reason=0000, extension=00000000
BUFFER RLSE  call. Return code=00000000, reason=00000000
ODBM DEREG   call. Return code=00000000, reason=00000000
SCI QUIESCE  call. Return code=00000000, reason=00000000
SCI DEREG    call. Return code=00000000, reason=00000000
```

C.3 Sample BPE DBRC statistics exit

File MYDSTAT0.ASM.TXT is an assembler BPE statistics exit for use with DBRC. It uses many IMS macros and so you will need the SDFSMAC library to assemble it.

This program is not intended as an example of good programming, merely to illustrate how you might write such an exit.

The program reports (through WTO) a simple one line summary each time the online DBRC region drives the exit. This summary contains:

- Total number of DBRC calls made by the client
- Average elapsed time (in milliseconds)
- Maximum elapsed time of any such call (also in milliseconds)

We have chosen to report only a few fields. There are many more fields that the exit could report on. For details about the BPE Statistics user-supplied exit routine, refer to *IMS Version 11 Exit Routines,* SC19-2437.

The messages look like Example C-5 on page 333, if you have let STATINTV default, the exit will report every 10 minutes:

Example C-5 Sample messages from program MYDSTAT0

```
19.57.22 ... Client IM2B          46 calls. Wait   16.213 (avg)   85.015 (max).
20.07.22 ... Client IM2B          51 calls. Wait   16.213 (avg)   85.015 (max).
20.17.22 ... Client IM2B          52 calls. Wait   16.213 (avg)   85.015 (max).
20.27.22 ... Client IM2B          68 calls. Wait    7.855 (avg)   89.122 (max).
```

If you reassemble MYDSTAT with SYSPARM(Y), the exit will also report the breakdown by DBRC call type. This looks like Example C-6.

Example C-6 Sample detailed messages from program MYDSTAT0

```
Client IM2B          28 calls. Wait   11.268 (avg)  145.920 (max).
       OPEN           1 calls. Wait   48.018 (avg).
       AUTH           4 calls. Wait   31.045 (avg).
       SIGNON         2 calls. Wait   20.485 (avg).
       SIGNON R       1 calls. Wait    6.069 (avg).
       SIGNOFF        1 calls. Wait   20.210 (avg).
       DBOPEN         4 calls. Wait   13.441 (avg).
       /RM CMD        3 calls. Wait   50.120 (avg).
       XDLI QRY       1 calls. Wait   22.224 (avg).
       SWITCH         1 calls. Wait   14.911 (avg).
       STATUS         3 calls. Wait   18.973 (avg).
       LAST           2 calls. Wait   13.542 (avg).
       DB INFO        1 calls. Wait   12.132 (avg).
       QSC STA        1 calls. Wait   38.153 (avg).
       QSC HOLD       1 calls. Wait   32.061 (avg).
       QSC END        1 calls. Wait  145.911 (avg).
       QSC UPTB       1 calls. Wait    5.724 (avg).
```

This is quite verbose and so you would not normally use this option. But, because you can refresh the exit while DBRC is running, it would be relatively easy to switch between the two versions of the module.

D

Additional material

This book refers to additional material that can be downloaded from the Internet as described in this section.

Locating the Web material

The Web material associated with this book is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser at:

`ftp://www.redbooks.ibm.com/redbooks/SG247807`

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG247807.

Using the Web material

The additional Web material that accompanies this book includes the following compressed files (in `.zip` file format):

File name	Description
`DRDA1.zip`	Sample z/OS DRDA program; refer to C.1, "Sample DRDA program" on page 326.
`MYDSTAT0.zip`	Sample DBRC statistics exit; refer to C.3, "Sample BPE DBRC statistics exit" on page 332.
`ODBM1.zip`	Sample ODBM client; refer to C.2, "Sample ODBM program" on page 331.
`OpenDBSamples.zip`	Code samples for the Windows ODBM examples; refer to B.2, "ODBM sample programs" on page 323.

System requirements for downloading the Web material

The following system configuration is suggested:

- Hard disk space: 2 MB minimum
- Operating System: Windows
- Processor: Intel® 386 or higher
- Memory: 16 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and extract the contents of the Web material `.zip` file into this folder.

Abbreviations and acronyms

ACEE	access control environment element
AGN	Application Group Name
AIB	Application Interface Block
AOI	Automated Operator Interface
APF	authorized program facility
API	application programming interface
APPC	Advanced Program-to-Program Communication
BMP	batch messaging program
BPE	Base Primitive Environment
CBM	Component Business Modeling
CCF	Common Connector Framework
CDE	contents directory entry
CGI	Common Gateway Interface
CI	Control Interval
CICS	Customer Information Control System
CLS	Common Service Layer
CSM	Complete Status Message
CTG	CICS Transaction Gateway
DB2	Database 2
DBCTL	Database Control
DBD	database description
DBRA	Database Resource Adapter
DBRC	Database Recovery Control
DEDB	data entry database
DMB	data management block
DRDA	Distributed Relational Database Architecture
EAB	Enterprise Access Builder
ECB	Event Control Block
EJB	Enterprise JavaBeans
ETO	Extended Terminal Option
EWLM	Enterprise Workload Manager
GUI	graphical user interface
HALDB	High Availability Large Data Bases
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines Corporation
ILDS	indirect list data set
IMS	Information Management System
IPCS	Interactive Problem Control System
IPL	initial program load
IRLM	internal resource lock manager
IRM	IMS request message
ISC	intersystem communication
ISPF	Interactive Systems Productivity Facility
ITSO	International Technical Support Organization
IVP	Installation Verification Program
J2C	Java EE Connector Architecture
Java EE	Java Platform Enterprise Edition
JBP	Java Batch Program
JCA	Java EE Connector Architecture
JCL	job control language
JDBC	Java Database Connectivity
JDK	Java Development Kit
JMP	Java Message Program
JRE	Java SE Runtime Environment
JSP	JavaServer Pages
JVM	Java virtual machine
KBLA	Knowledge-Based Log Analysis
LAN	local area network
LPAR	logical partition
LSQA	local system queue area
LTERM	logical terminal
LU	logical unit
LU2	logical unit 2
MCI	Message Control Information
MDB	message-driven bean
MFS	Message Format Service
MOD	message output descriptor
MPP	message processing program
MSC	Multiple Systems Coupling
MVS	Multiple Virtual System
ODBA	Open Database Access
ODBM	Open Database Manager
OLDS	online log data set
OLR	Online Reorganization
OM	Operations Manager

OO	object-oriented
OTMA	Open Transaction Manager Access
OTMA C/I	OTMA Callable Interface
PC	personal computer
PRA	Parallel RECON Access
PCB	program communication block
PPT	Program Properties Table
PSB	program specification block
RACF	Resource Access Control Facility
RAR	resource adapter archive
RDDS	Resource Definition Data Set
RECON	Recovery Control
RM	resource manager
RMM	Request MOD Message
RRS/MVS	Resource Recovery Services/MVS
RSM	request status message
SAF	System Authorization Facility
SCI	Structured Call Interface
SGML	Standard Generalized Markup Language
SLSB	stateless session bean
SMP/E	System Modification Program/Extended
SMU	Security Maintenance Utility
SNA	Systems Network Architecture
SOA	service-oriented architecture
SOMA	Service Oriented Modeling and Architecture
SPE	small programming enhancement
SPOC	single point of control
SSPM	Sysplex serialized program management
STE	storage tracking element
STSN	set and test sequence numbers
SVL	Silicon Valley Laboratories
TCO	Time-Controlled Operations
TCP/IP	Transmission Control Protocol/Internet Protocol
TMRA	Transaction Manager Resource Adapter
TPIPE	transaction pipe
UOR	Unit of Recovery
VIPA	Virtual IP Addressing
W3C	World Wide Web Consortium
WAN	wide area network
WLM	workload manager
WSDL	Web Service Description Language
WWW	World Wide Web
XCF	cross-system coupling facility
XMI	XML Metadata Interchange
XML	Extensible Markup Language

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 341. Note that some of the documents referenced here may be available in softcopy only.

- *Powering SOA Solutions with IMS*, SG24-7662
- *Powering SOA with IBM Data Servers,* SG24-7259
- *IMS Performance and Tuning Guide,* SG24-7324
- *IMS V10 Implementation Guide: A Technical Overview,* SG24-7526
- *IMS Connectivity in an On Demand Environment: A Practical Guide to IMS Connectivity,* SG24-6794
- *Publishing IMS and DB2 Data Using WebSphere Information Integrator: Configuration and Monitoring Guide,* SG24-7132
- *IMS V9 Implementation Guide: A Technical Overview,* SG24-6398
- *IMS V8 Implementation Guide: A Technical Introduction of the New Features,* SG24-6594
- *Reorganizing Databases Using IMS Tools: A Detailed Look at the IBM IMS High Performance Tools,* SG24-6074
- *IMS Installation and Maintenance Processes,* SG24-6574
- *Using IMS Data Management Tools for Fast Path Databases,* SG24-6866
- *IMS in the Parallel Sysplex Volume I: Reviewing the IMSplex Technology,* SG24-6908
- *IMS in the Parallel Sysplex Volume II: Planning the IMSplex,* SG24-6928

Other publications

These publications are also relevant as further information sources:

- *IMS Version 11 Application Programming,* SC19-2428
- *IMS Version 11 Application Programming APIs,* SC19-2429
- *IMS Version 11 Commands, Volume 1: IMS Commands A-M,* SC19-2430
- *IMS Version 11 Commands, Volume 2: IMS Commands N-V,* SC19-2431
- *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands,* SC19-2432
- *IMS Version 11 Communications and Connections,* SC19-2433
- *IMS Version 11 Database Administration,* SC19-2434
- *IMS Version 11 Database Utilities,* SC19-2435
- *IMS Version 11 Diagnosis,* GC19-2436

- *IMS Version 11 Exit Routines,* SC19-2437
- *IMS Version 11 Installation,* GC19-2438
- *IMS Version 11 Licensed Program Specifications,* GC19-2439
- *IMS Version 11 Master Index and Glossary,* SC19-2440
- *IMS Messages and Codes, Volume 1: DFS Messages,* GC18-9712
- *IMS Messages and Codes, Volume 2: Non-DFS Messages,* GC18-9713
- *IMS Messages and Codes, Volume 3: IMS Abend Codes,* GC18-9714
- *IMS Messages and Codes, Volume 4: IMS Component Codes,* GC18-9715
- *IMS Version 11 Operations and Automation,* SC19-2441
- *IMS Version 11 Release Planning,* GC19-2442
- *IMS Version 11 System Administration,* SC19-2443
- *IMS Version 11 System Definition,* GC19-2444
- *IMS Version 11 System Programming APIs,* SC19-2445
- *IMS Version 11 System Utilities,* SC19-2446
- *IMS Version 11 Fact Sheet,* GC19-2451
- *IMS Version 11 Program Directory for Information Management System Transaction and Database Servers V11.0,* GI10-8788
- *IRLM Messages and Codes for IMS and DB2 for z/OS,* GC19-2666
- *IMS Enterprise Suite Program Directory,* GI10-8816
- *IMS Enterprise Suite License Information,* GC19-2807

Online resources

These Web sites are also relevant as further information sources:

- IMS products and tools

  http://www.ibm.com/ims

  This Web site also contains the following materials, which are available in PDF, HTML, or PowerPoint format:

  – IMS: An Overview Presentation Guide
  – IMS Version 11 Presentation Guide
  – IMS: What's New and What's Next
  – IMS On Demand Integrated Solutions
  – IMS V10: What's New Since GA
  – IMS Update
  – SOA Access to and From IMS
  – IMS White papers on SOA, XML, Performance, Connectivity, and so on

- IMS Information Center

  http://publib.boulder.ibm.com/infocenter/imzic

- IMS Version 11 QPP Information Center

  http://dzictestsrv.svl.ibm.com:8035/help/index.jsp

- The IMS Enterprise Suite set of topics

  http://publib.boulder.ibm.com/infocenter/imzic/topic/com.ibm.ims.es.doc/ies_home.htm
- System z hardware

  http://www.ibm.com/systems/z/hardware/
- IBM DB2 and IMS tools expanded and enhanced to help better manage your environments, Announcement letter US 209-353

  http://www-306.ibm.com/common/ssi/fcgi-bin/ssialias?infotype=an&subtype=ca&appname=xldata&htmlfid=897/ENUS209-353

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks publications, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM

Redbooks

IMS Version 11 Technical Overview

(0.5" spine)
0.475"<->0.873"
250 <-> 459 pages

IMS Version 11 Technical Overview

Explore IMS 11 functions

Understand advantages and applicability

Plan for installation or migration

IMS provides leadership in performance, reliability and security to help you implement the most strategic and critical enterprise applications. IMS also keeps pace with the evolving IT industry.

Information Management System Version 11 (IMS 11) provides an open, integrated, and distributed data access solution; improves connectivity and ease of use; increases system availability; and offers an architectural road map that supports future growth.

With IMS 11, a suite of Universal drivers supports IMS database programmatic access with Type-4 and Type-2 connectivity. You can now access IMS in a uniform way using the industry standard Distributed Relational Database Architecture (DRDA) protocol from any platform and from within the most strategic run times, opening growth and expansion opportunity. IMS DB metadata is exposed with the standard JDBC API and therefore can be consumed and visualized by JDBC tooling. Query syntax uses standard query language syntax.

The new IMS Enterprise Suite V1.1 components are designed to enhance your use of IMS applications and data by delivering innovative capabilities that enhance connectivity, expand application development, extend standards and tools for a service-oriented architecture (SOA), ease installation, and provide simplified interfaces.

IMS 11 also helps lower your IT costs in the areas of business flexibility, simplified administration, and growth.

In this IBM Redbooks publication, we explore the new features of IMS 11. We review the available material and include feedback from early users. The intent is to highlight the major new functions and facilitate installation and migration.

SG24-7807-00 ISBN 0738433578