IMS Version 11

Information Management software

Class Agenda

- Introduction
  - Agenda
  - Prerequisites
- System Enhancements
  - DRD Export, Import, and Utilities
  - ACBLIB Usability Enhancements
  - SPOC Print Options (IMS 10 SPE)
  - GSAM XRST Enhancement
  - U0845 Diagnostics
  - /DIAGNOSE Command Enhancements
  - Dump Formatter Enhancement
  - Dynamic Abend Dump Exit
  - LSQA Storage Reduction

Class Agenda

- System Enhancements (cont.)
  - Extended Address Volumes (EAV) Support
  - New User Exits
  - KBLA Enhancements
- Security
  - RACF Mixed-case Password Default
  - SMU to RACF Conversion Aid Utilities
- Database
  - Database Quiesce
  - HALDB Online Reorg
  - Dynamic Locktime
  - Fast Path 64-bit Buffer Manager
  - Fast Path Database Usability and Serviceability

Class Agenda

- IMS Tools Update
- Transaction Manager
  - Type-2 Query Commands for DC Resources
  - DFSMSCE0 Enhancements
  - Full Function Response Mode Recovery
  - Shared Queues False Scheduling Reduction
  - Transaction Expiration
- Connectivity
  - OTMA
    - Consistency enhancements for Shared Queues environments
    - Timeout enhancements
    - Resource Monitoring
    - Usability

Class Agenda

- Connectivity (cont.)
  - IMS Connect
    - IMS Connect configuration member HWSCFGx enhancements
    - Enhanced commands
    - Exit Routine enhancements
    - Cancel Client ID
    - TCP/IP Auto Reconnect
    - Generated Client ID
    - Performance enhancement
    - New Recorder Trace and BPE support

Class Agenda

- Integration
  - IMS Web 2.0 Solution for InfoSphere MashupHub
  - IMS TM Resource Adapter 10.2
  - IMS 10 Synchronous Callout
  - IMS 10 Database Web Services
  - IMS 10 Soap Gateway
  - IMS 11 Open Database
- IMS Enterprise Suite
  - SOAP Gateway
  - IMS Connect API
  - DLIModel Utility
  - JMS API for Synchronous Callout

Class Agenda

- DBRC
  - BPE Based DBRC Region
  - Unconditional Deletion of RECON Information
  - Security Override for Non-production RECON Copies
  - Change Accumulation Usability
  - DELETE.DB Performance
  - DBRC Migration
- Installation and Migration
  - Packaging, Prerequisites, and Coexistence
  - IMS Library Changes
  - IVP and Syntax Checker
  - Installation and Migration Tasks
  - Review of Migration Considerations

Software Prerequisites

- Minimum software level prerequisites
  - z/OS V1R9 (5694-A01)
    - RACF, or equivalent, if security is used
    - High Level Assembler Toolkit Release 5 (5696-234)
  - IRLM 2.2, if IRLM is used
- Minimum software levels for optional functions:
  - The DLIModel utility plug-in
    - Requires Eclipse version 3.2.2 and either Rational Developer for System z (RDZ), version 7.1.0 or Rational Application Developer (RAD), version 7.0.0.4
  - Java requires SDK 6.0
  - Parallel RECON Access (introduced in IMS 10) requires Transactional VSAM
    - Special bids (price discounts) are considered for Transactional VSAM
  - See the IMS 11 *Release Planning* publication for additional requirements

8

The minimum level of z/OS for IMS 11 is z/OS V1R9. In addition to z/OS the user must install RACF, or an equivalent security product, in order to use security with IMS 11. RACF is part of the SecureWay Security Server. As with previous IMS releases, the High Level Assembler Toolkit is required to provide assembler macros that IMS uses. If the IRLM is used, IRLM 2.2 is required. Program Isolation (PI) is also supported with IMS 11. IRLM is required for block level data sharing.

The DLIModel utility is an Eclipse plug-in. It requires Eclipse version 3.2.2 with either Rational Developer for System z (RDZ), version 7.1.0 or Rational Application Developer (RAD), version 7.0.0.4.

The use of Java with IMS 11 requires SDK 6.0.

The IMS 11 *Release Planning* publication has additional information about requirements when using particular functions in IMS 11. This is especially important for Java users. The level of JDK or SDK depends on the environment in which Java will execute.

Supported Migrations

- IMS 10 to IMS 11
  - Upgrade RECONs from IMS 10 to IMS 11
  - Databases are compatible
  - Application programs are compatible
- IMS 9 to IMS 11
  - Upgrade RECONs from IMS 9 to IMS 11
  - Databases are compatible
  - Application programs are compatible

9

The details for installing IMS 11 and migrating systems from IMS 10 and IMS 9 to IMS 11 are discussed in the Installation and Migration section of the class. This will include an explanation of the Small Program Enhancements (SPEs) for IMS 10 and IMS 10 that provide compatibility during the migration process.

Databases and application programs do not have to be modified, upgraded, recompiled, or relinked for use with IMS 11. Those used with IMS 9 and IMS 10 and earlier IMS releases are compatible with IMS 11.

IMS Version 11

System

Information Management software

System Enhancements

- DRD Export, Import, and Utilities
- ACBLIB Usability Enhancements
- SPOC Print Options (IMS 10 SPE)
- GSAM XRST Enhancement
- U0845 Diagnostics
- /DIAGNOSE Command Enhancements
- Dump Formatter Enhancement
- Dynamic Abend Dump Exit
- LSQA Storage Reduction
- Extended Address Volumes (EAV) Support
- New User Exits
- KBLA Enhancements

Dynamic Resource Definition EXPORT/IMPORT (IMS 10 SPE)

3

The Dynamic Resource Definition (DRD) EXPORT/IMPORT commands are delivered via maintenance for IMS 10. The EXPORT command via APARs PK66704 and PK89893 and PTF UK41049. The IMPORT command via APARs PK66682 and PK89893 and PTFs UK41047 and UK48863.

Resource Definition Data Set Terminology

- BSAM data set used to save MODBLKS resource and descriptor definitions
- System RDDS
  - Provides a single system view of an IMS's resources and descriptors
  - Contains all resource and descriptor definitions for an IMS
  - Each IMS must define its own set of system RDDS data sets
- Non-System RDDS
  - Can be shared between IMSs
  - May contain a subset of an IMS’s resource and descriptor definitions

4

Before discussing the DRD IMPORT and EXPORT commands, here is some terminology that will be used throughout the next several slides. The Resource Definition Data Set (RDDS) is the data set used with the DRD capability to contain the MODBLKS resource and descriptor definitions. The details of this data set are not covered here, since we will only focus on the EXPORT/IMPORT commands. Rather, the purpose of this slide is to clarify the difference between a “system RDDS” and a “non-system RDDS”.

A system RDDS is defined in an IMS system’s DFSDFxxx PROCLIB member on the RDDSDSN= parameter. It contains all of the MODBLKS resource/descriptor definitions for that particular IMS. Therefore, a set of system RDDSs is unique to a single IMS system.

A non-system RDDS can be shared among multiple IMS systems and can contain definitions from different IMS systems. Therefore, it may contain a subset of an IMS’s MODBLKS resource/descriptor definitions.

EXPORT Command

- Delivered in APAR PK66704 and PK89893; PTF UK41049
- Used to export resource and descriptor definitions to an RDDS
- EXPORT Command Syntax

```
EXPORT DEFN TARGET(RDDS) TYPE() NAME() RDDSDSN() OPTION()
```

- Where
  - **TYPE()** defines the resource type
    - ALL, ALLDESC, ALLRSC ,DB, DBDESC, PGM, PGMDESC, RTC, RTCDESC, TRAN, TRANDESC
  - **NAME()** defines the names of the resources to export
    - NAME(*) is the default

5

A new Type-2 command EXPORT was added as an IMS 10 SPE, that allows for the exporting of MODBLKS resources and descriptor definitions to either a system or non-system RDDS. The command syntax is listed in the green box and here is some more detail on each of the parameters:

DEFN: Specifies descriptor and resource definitions.

TARGET(RDDS): Specifies that resources and descriptors are to be exported to an RDDS data set.

TYPE(): Specifies the type of resource or descriptor to export. This is an optional parameter for TARGET (RDDS) and multiple TYPEs can be listed in the same command. The default value is ALL, meaning all resources and descriptors types. Other more granular types that reference a smaller subset of resources/descriptors can be specified as well, for example “all descriptors”, “all resources” and/or individual resource/descriptor types.

NAME(): Specifies the 1-8 character name of a resource or descriptor. For NAME(*) which is also the default value, command responses are returned only for the resource and descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all of the resource and descriptor names that are processed.

EXPORT Command

`EXPORT DEFN TARGET(RDDS) TYPE() NAME() RDDSDSN() OPTION()`

- **RDDSDSN()** defines a non-system resource definition data set
  - Cannot be a system RDDSs defined in DFSDFxxx member
    - To EXPORT to a system RDDS, omit this parameter and it will be selected as the target RDDS
- **OPTION()**
  - **OVERWRITE (default)** indicates that resource/descriptor definitions are to overwrite all existing definitions in the RDDS
    - Valid when exporting to system and non-system RDDS
  - **APPEND** indicates that resource/descriptor definitions are to be appended to the end of the specified RDDS data set
    - Valid when exporting to non-system RDDS

6

Here is a continuation of the list of EXPORT parameters:

RDDSDSN(dsname)

Specifies an optional data set to export resources and descriptors to. This data set cannot be one of the data sets defined on the RDDSDSN= parameter in the DFSDFxxx PROCLIB member (in other words, it cannot be a system RDDS). If the desired target RDDS is the system RDDS, omit this parameter from the command and it will be defaulted to. If there are no system RDDSs defined in an IMS's DFSDFxxx PROCLIB member, then RDDSDSN() is required on the EXPORT command. If using the RDDSDSN() parameter to specify a specific non-system RDDS, care should be taken to route the EXPORT command to only one IMS using OM routing. If the command is routed to multiple IMSs in the IMSplex, each IMS will export to the same non-system RDDS and the last IMS will overwrite all prior definitions.

OPTION()

OVERWRITE: Indicates that the resource and descriptor definitions are to overwrite all existing definitions in the RDDS. APPEND is mutually exclusive with OVERWRITE. OVERWRITE is the default and between the two and can be used when exporting to both a system and non-system RDDS.

APPEND: Indicates that the resource and descriptor definitions are to be appended to the end of the specified RDDS data set. If OPTION(NOCHECK) is not specified, validity checking is done to ensure the IMS performing the export is the same IMS that previously exported definitions to the RDDS. The APPEND option can only be used when exporting to a non-system RDDS. APPEND is mutually exclusive with OVERWRITE.

EXPORT Command

```
EXPORT DEFN TARGET(RDDS) TYPE() NAME() RDDSDSN() OPTION()
```

- **OPTION()**
  - **NOCHECK** indicates that no ID checking is to be done when definitions are appended to an RDDS and that the IMS performing the export can be a different IMS than the one that performed the previous export.
    - Only valid when exporting to a non-system RDDS
  - **ALLRSP** returns a response for each exported resource, valid with NAME(*) only

7

Here is a continuation of the list of EXPORT parameters:

OPTION()

NOCHECK: Indicates that no ID checking is to be done when definitions are appended to an RDDS. If OPTION(NOCHECK) is specified, the IMS performing the export can be a different IMS than the one that performed the previous export. The NOCHECK option is only valid when exporting to a non-system RDDS. NOCHECK is mutually exclusive with OVERWRITE and can only be specified if APPEND is specified.

ALLRSP: Indicates that response lines are to be returned for all resources and descriptors that are processed by the command. If NAME(*) is specified, the default action is to return response lines only for the resources and descriptors that resulted in an error. OPTION(ALLRSP) can be used with NAME(*) to request response lines for all resources and descriptors processed by the command. If a name value other than NAME(*) is specified, the default action is to return a response line for all resources and descriptors processed by the command. If OPTION(ALLRSP) is specified with a name value other than NAME(*), the ALLRSP parameter is ignored.

For HALDB databases, only the definition of the HALDB master database is exported. HALDB partition definitions are not exported, because the partition definitions are maintained in the RECON data set and not defined with CREATE commands or DATABASE system definition macros. OPTION(ALLRSP) does not include the partitions definitions in the output.

EXPORT Command Considerations

- The EXPORT DEFN command can be issued in both a DRD and non-DRD environment
  - In a non-DRD environment the export must be to a non-system RDDS and only resources (no descriptors) will be exported
- Command routing depends on RDDS type
  - System RDDS
    - All IMS systems that receive the command will export resources/descriptors to its own system RDDS
  - Non-system RDDS
    - Only one command master IMS will export the specified resources/descriptors to the non-system RDDS listed on the RDDSDSN() parameter

8

The EXPORT command can be issued to a non-DRD enabled IMS system. However, in a non-DRD environment, the only RDDS that can be exported to is a non-system RDDS.

Also, the command will be routed differently depending on what type of RDDS is being targeted when it is issued. In the case of a system RDDS (no RDDSDSN() parameter specified on the command), each IMS system that the command is routed to (default routing is all but a specific route list can be used as well) will export its resource/descriptor definitions to its own RDDS specified in its DFSDFxxx PROCLIB member.

If RDDSDSN() is specified on the EXPORT command, a non-system RDDS will be used in which case only one command master IMS will process the command. This is to ensure that there is not data contention.

EXPORT Command Considerations

- When exporting to a system RDDS
  - All resource and descriptor definitions must be exported
  - Existing definitions are overwritten
  - ! Recommendation: issue EXPORT before shutting down IMS to preserve timestamps across coldstart
- When exporting to a non-system RDDS
  - A subset of resource and descriptor definitions may be exported
  - Existing definitions can be overwritten or appended to
  - Important tip: when routing successive EXPORT commands to each system in an IMSplex and RDDSDSN() is specified, include OPTION(APPEND)

9

In the case of exporting definitions to a system RDDS, all resource/descriptor definitions must be exported and will overwrite the definitions that previously existed in the system RDDS.

On the other hand, when exporting to a non-system RDDS, there is more flexibility in that a subset of resource/descriptor definitions are able to be exported and the definitions that previously existed in the non-system RDDS can be preserved. There is still the option to overwrite them as well.

If the EXPORT command is issued multiple times, once for each IMS system in an IMSplex, specifying the same non-system RDDS it is advisable to include the OPTION(APPEND) parameter on the command. This will ensure that each command will not overwrite the data written from a previous command.

EXPORT Command Restrictions

- EXPORT command not supported for the following IMS types
  - XRF alternate
  - RSR tracker
  - FDBR region
- IMS defined resources cannot be exported
  - DBF#FPU0,
  - DBFDSRT1, DFSDSDB1, DFSDSPG1, DFSDSTR1

10

There are a few restrictions associated with the EXPORT command. The EXPORT command is not able to be issued for an XRF alternate IMS, an RSR tracker or an FDBR region. Additionally, certain IMS-defined resources are not able to be exported and they are listed here.

IMPORT Command

- Delivered in APARs PK66682 and PK89893 and PTFs UK41047 and UK48863
- Used to import resource and descriptor definitions from an RDDS
- IMPORT Command Syntax

```
IMPORT DEFN SOURCE(RDDS) TYPE() NAME() RDDSDSN() OPTION()
```

- Where
  - **TYPE()** defines the resource type
    - ALL, ALLDESC, ALLRSC, DB, DBDESC, PGM, PGMDESC, RTC, RTCDESC, TRAN, TRANDESC
  - **NAME()** defines the names of the resources to import
    - NAME(*) is the default

11

A new Type-2 command IMPORT was added as an IMS 10 SPE, that allows for the importing of MODBLKS resources and descriptor definitions from either a system or non-system RDDS. The command syntax is listed in the green box and here is some more detail on each of the parameters:

DEFN: Specifies descriptor and resource definitions.

SOURCE(RDDS): Specifies where resources and descriptors are to be located for importing. RDDS specifies that resources and descriptors are to be imported from an RDDS.

TYPE(): Specifies the type of resource or descriptor to import. This is an optional parameter for SOURCE (RDDS) and multiple TYPEs can be listed in the same command. The default value is ALL, meaning all resources and descriptors types. Other more granular types that reference a smaller subset of resources/descriptors can be specified as well, for example “all descriptors”, “all resources” and/or individual resource/descriptor types.

NAME(): Specifies the 1-8 character name of a resource or descriptor. NAME() supports individual unique names and wildcards for non-system RDDS. For a system RDDS, NAME(*) is the only valid value. For the IMPORT command, NAME(*) is the default value and command responses are returned only for the resource and descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all of the resource and descriptor names that are processed.

IMPORT Command

```
IMPORT DEFN SOURCE(RDDS) TYPE() NAME() RDDSDSN() OPTION()
```

- **RDDSDSN()** defines either a system or non-system resource definition data set
  - If omitted, imports from a system resource definition data set
- **OPTION()**
  - ALLRSP returns a response for each imported resource, valid with NAME(*) only
  - ABORT imports no resources or descriptors at all in the event of an error

12

Here is a continuation of the list of IMPORT parameters:

RDDSDSN(dsname): Specifies an optional data set from which to import the descriptor and resource definitions. This data set can be either a system or non-system RDDS. If RDDSDSN() is not specified, the resources and descriptors are imported from the newest RDDS by the RDDSDSN parameter in the DFSDFxxx PROCLIB member. If RDDDSN is not specified in the DFSDFxxx PROCLIB member, then RDDSDSN() is required.

OPTION(): Specifies additional functions to be performed along with the command.

ALLRSP

Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid when NAME(*) is specified or defaulted to.

ABORT

Processes the definitions in the data set as a group. If an error occurs in any of the entries, the IMPORT command fails and none of the descriptors or resources are imported.

IMPORT Command Considerations

- If duplicate resources or descriptors exist in an RDDS, the newest instance will be used
  - Could arise when importing from a non-system RDDS that contains multiple resource definitions for the same resource
- IMPORT supports CREATE of new MODBLKS resources only
  - Does not support UPDATE existing MODBLKS resources
- When importing from a system RDDS, TYPE(ALL) and NAME(*) will always be used (if anything else specified for these parameters, command will fail)

IMPORT Command Considerations

- HALDB databases: can import HALDB Master only, no partitions
  - Successful import of HALDB Master will result in ‘ D’ completion code with accompanying text returned:

```
RSC NEEDS STA DB ON HALDB MASTER
```

    - Serves as a reminder that HALDB partitions must be built with either /STA DB command or UPDATE DB START(ACCESS)

There are a few restrictions for the IMPORT command. IMPORT can only be used to create new MODBLKS resources. It does not allow for the updating of resources. So for example, PGM1 is updated with the UPDATE command on IMSA and EXPORTed to an RDDS. IMSB cannot IMPORT PGM1 if it already has a PGM1 defined in its online system.

For HALDB databases, IMPORT can only be used to import HALDB Master definitions but not HALDB partitions. When a HALDB Master is successfully imported, a completion code of ‘ D’ will be returned with some informational text indicating that the partitions need to be built in order to be brought online. This can be accomplished with either the /START DB command or the UPDATE DB START(ACCESS) command.

Normally, HALDB partitions are read from the RECON during IMS initialization and brought online (control blocks are built) as part of the coldstart process. Since we’re now dynamically importing the HALDB Master, the /START DB or UPDATE DB START(ACCESS) commands need to be used to perform the building of the partitions.

EXPORT/IMPORT with IMS Application Migration

- Use EXPORT/IMPORT to migrate an application defined on one IMS system to another IMS system
  - Issue EXPORT command on IMSA to export an application's database, program, routing code and transaction definitions to a non-system RDDS
  - Issue IMPORT command on IMSB to import the definitions

! - Use EXPORT to migrate definitions from an IMS system (DRD enabled or not) to other DRD-enabled IMS systems

15

Using the IMPORT and EXPORT commands can be useful in application migration. These commands can be used to migrate an application's resources from one IMS system to another. For example, the EXPORT command can be issued on IMSA to export the application's resources to a non-system RDDS and IMSB can subsequently import those definitions using the IMPORT command. Note that a non-system RDDS must be used in this case.

The commands can also be used to migrate resource definitions from an IMS system to systems that are DRD-enabled. The system from which the definitions are exported may either be DRD-enabled or not DRD-enabled.

EXPORT/IMPORT with IMS Cloning

- Use EXPORT/IMPORT to clone IMS systems
- Example steps
  1. EXPORT definitions from IMSA to non-system RDDS
  2. Coldstart new IMSB with no resources defined
  3. IMPORT definitions to IMSB from non-system RDDS

16

In the same way, the EXPORT/IMPORT commands can be used in the cloning of IMS systems. For example, the EXPORT command can be used to export IMSA's definitions to a non-system RDDS. IMSB can be coldstarted with no resources defined and can IMPORT IMSA's definitions from the non-system RDDS with the command.

Benefits

- EXPORT/IMPORT commands
  - Enable capability of dynamically porting MODBLKS resource/descriptor definitions between IMS systems, useful in:
    - IMS cloning
    - Application migration

The IMPORT and EXPORT commands are useful because when used in conjuction with one another, they enable the portability of MOBDLKS resources/descriptors between IMS systems. This is especially helpful in the areas of IMS cloning and application migration.

Dynamic Resource Definition Utilities

(IMS 10 SPE)

RDDS Extraction Utility (DFSURDD0) - Enhancement

- Offline batch utility
- Converts stored resource definitions in an RDDS to:
  - IMS Stage-1 macro statements (APPLCTN, TRANSACT, DATABASE, RTCODE)
  - IMS Type-2 CREATE commands for DRD (could be submitted to IMS via batch SPOC)
    - CREATE DB; CREATE TRAN; CREATE PGM ; CREATE RTC
    - CREATE DBDESC; CREATE TRANDESC; CREATE PGMDESC; CREATE RTCDESC
- Output written to SYSOUT data set; messages written to SYSPRINT
- Enhancement made by IMS 10 SPE (PK63186; UK43484):
  - Support added to write resources/descriptors with all attribute values
    - OUTPUT=QUERY keyword
  - Also reports duplicates

19

IMS 10 included the RDDS Extraction utility. This utility (DFSURDD0) runs offline and reads either a system or non-system RDDS to convert the contents into either Stage 1 macro statements or IMS Type-2 CREATE commands. When this utility is run, the output is written to the SYSOUT data set. Messages are written to the SYSPRINT data set.

The IMS 10 SPE allows querying the RDDS to display its contents, which will be resources and descriptor definitions along with their attribute values. Duplicates are flagged as well. This enhancement was delivered via APAR PK63186 and PTF UK43484.

New DRD Utilities

- IMS 10 SPE (APARs PK71567 and PK86125; PTF UK43484)
- Set of offline utilities to perform the following DRD-related functions:
  - Generate RDDS from
    - Checkpoint and X'22' (type-2 command) log records
    - MODBLKS data set
    - SYSGEN macros
  - Generate CREATE commands from
    - MODBLKS data set
    - SYSGEN macros
  - Display RDDS content
- 'Manage RDDS' option added to IMS Manage Resources application

20

In addition to the RDDS Extraction Utility, another set of DRD utilities was created which are contained in APAR PK71567 and PK86125 (PTF UK43484) for IMS 10. These utilities will also run offline.

DRD Usability enhancements include:

- Build of RDDS from checkpoint and X'22' log records
- Build of RDDS from the MODBLKS data set
- Build of RDDS from Stage 1 input
- Generate CREATE commands from the MODBLKS data set
- Generate CREATE commands from Stage 1 input macros
- Writing the RDDS contents
- The 'Manage RDDS' option is added as a selection from the 'IMS Manages Resources' panel within the IMS Application menu.

New DRD Utilities

- Utilities run offline and are invoked via JCL streams
- JCL can be generated by either:
  - ISPF Panels accessible from the Manage Resources application off the IMS Application Menu
  - Utilities are documented in the *IMS System Utilities Reference* manual
    - Use of the ISPF panels are not required in order to invoke utilities

21

The DRD utilities are invoked via JCL streams, which can be automatically generated via the Manage Resources application or written using the IMS System Utilities Reference as a guide.

Benefits

- DRD Utilities
  - Facilitate implementation of DRD by reducing manual effort required to create an RDDS
  - Make DRD more usable by providing capability to re-create an “up-to-date” RDDS in the event of accidental loss
    - Allow creation of an RDDS from IMS system definitions, MODBLKS data set, checkpoint log records and X’22’ log records
    - Allow contents of an RDDS to be queried/displayed

The DRD Utilities are useful in that they facilitate the implementation of DRD by providing a way to create an RDDS with significantly less effort that would otherwise be necessary. They also make the DRD capability more useable usable by providing a way to re-create an RDDS in the event of an accidentally loss. Using the utilities, an RDDS can be re-created from IMS SYSGEN, MODBLKS, or either checkpoint or X’22’ log records. DRD is also made more useable by the utilities providing a way to query the contents of an RDDS.

Reference Section

EXPORT Command Examples

EXPORT Command Input/Output Example 1

TSO SPOC Input

```
EXPORT DEFN TARGET(RDDS) RDDSDSN(NON.SYS.RDDS1)
  TYPE(ALLDESC) OPTION(APPEND,ALLRSP)
```

TSO SPOC Output Response

Name	Type	MbrName	CC
PGMTEST	PGMDESC	IMS1	0
TEST	TRANDESC	IMS1	0

EXPORT Command Input/Output Example 2

TSO SPOC Input

```
EXPORT DEFN TARGET(RDDS) RDDSDSN(NON.SYS.RDDS1)
  TYPE(ALLRSC) NAME(TEST)
```

TSO SPOC Output Response

Name	Type	MbrName	CC
TEST	PGM	IMS1	0

EXPORT Command Input/Output Example 3

TSO SPOC Input

```
EXPORT DEFN TARGET(RDDS) RDDSDSN(NON.SYS.RDDS2)
  NAME(TEST,PGMTES*)
```

TSO SPOC Output Response

Name	Type	MbrName	CC
TEST	PGM	IMS1	0
PGMTEST	PGMDESC	IMS1	0
TEST	TRANDESC	IMS1	0

EXPORT Command Input/Output Example 4

TSO SPOC Input

```
EXPORT DEFN TARGET(RDDS) RDDSDSN(NON.SYS.RDDS1)
```

TSO SPOC Output Response

```
IMSplex . . . . . : PLEX1
 Routing . . . . . :
 Start time. . . . : 2008.294 15:55:10.77
 Stop time . . . . : 2008.294 15:55:10.82
 Return code . . . : 00000000
 Reason code . . . : 00000000
 Reason text . . . :
 Command master. . : IMS1

  MbrName  Messages
  IMS1     DFS3404I EXPORT COMMAND COMPLETE FOR ALL ,
  DSN=NON.SYS.RDDS1
```

EXPORT Command Input/Output Example 5

TSO SPOC Input

```
EXPORT DEFN TARGET(RDDS) RDDSDSN(NON.SYS.RDDS1) NAME(D*,
  E*, F*,HIMALM01,TPDYNCUR) TYPE(ALL) OPTION(OVERWRITE)
```

TSO SPOC Output Response

Name	Type	MbrName	CC
DBFSAMD2	DB	IMS1	0
EMHTX2	TRAN	IMS1	0
EMHTX3	TRAN	IMS1	0
FESTXA	TRAN	IMS1	0
FESTXB	TRAN	IMS1	0
HIMALM01	PGM	IMS1	0
TPDYNCUR	PGM	IMS1	0

29

IMPORT Command Examples

IMPORT Command Input/Output Example 1

TSO SPOC Input

```
IMPORT DEFN SOURCE(RDDS) OPTION(ABORT,ALLRSP)
```

TSO SPOC Output Response

```
IMSplex . . . . . . : PLEX1
 Routing . . . . . :
 Start time. . . . : 2008.294 15:55:04.40
 Stop time . . . . : 2008.294 15:55:04.41
 Return code . . . : 0200000C
 Reason code . . . : 00003004
 Reason text . . . : No requests were successful.
 Command master. . : IMS1
             Return     Reason
 MbrName     Code       Code       Description
 --------    ------     ------     -----------
 IMS1        00000008   0000211F   No system RDDSs defined,
  rddsdsn() required
```


IMPORT Command Input/Output Example 2

TSO SPOC Input

```
IMPORT DEFN SOURCE(RDDS) RDDSDSN(NON.SYS.RDDS1)
```

TSO SPOC Output Response

```
IMSplex . . . . . . : PLEX1
 Routing . . . . . :
 Start time. . . . : 2008.294 15:55:05.36
 Stop time . . . . : 2008.294 15:55:07.45
 Return code . . . : 00000000
 Reason code . . . : 00000000
 Reason text . . . :
 Command master. . : IMS1
 MbrName  Messages
 -------- -----------------------
 IMS1     DFS3405I IMPORT COMMAND COMPLETE FOR ALL ,
  DSN=NON.SYS.RDDS1
```

IMPORT Command Input/Output Example 3

TSO SPOC Input

```
IMPORT DEFN SOURCE(RDDS) RDDSDSN(IMSPLEX1.RDDS.DEFN)
  OPTION(ALLRSP)
```

TSO SPOC Output Response

Name	Type	MbrName	CC	ImpType
DEDBJN01	DB	IMS2	0	CREATE
DBDFLT	DBDESC	IMS2	0	CREATE
CDEBS	PGM	IMS2	0	CREATE
PGMDFLT	PGMDESC	IMS2	0	CREATE
SMQFP5A	RTC	IMS2	0	CREATE
RTCDFLT	RTCDESC	IMS2	0	CREATE
CDEBTRN1	TRAN	IMS2	0	CREATE
TRANDFLT	TRANDESC	IMS2	0	CREATE

IMPORT Command Input/Output Example 4

TSO SPOC Input

```
IMPORT DEFN SOURCE(RDDS) RDDSDSN(MYDSN.DEFN)
  OPTION(ABORT)
```

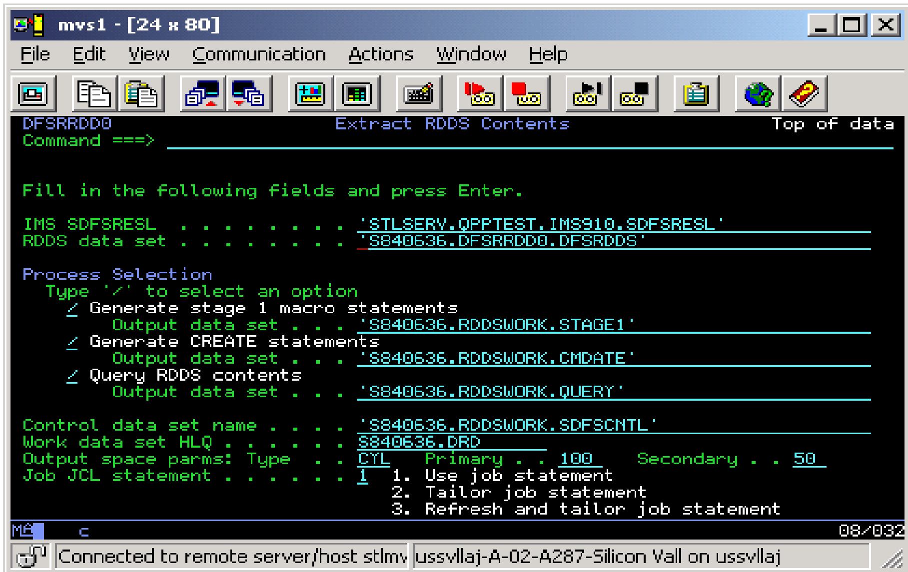
TSO SPOC Output Response

Name	Type	MbrName	CC	CCText
PART	DB	IMS1	9	NO IMPORT - OPTION ABORT
PARTMAST	DB	IMS1	9	NO IMPORT - OPTION ABORT
TESTDB	DB	IMS1	9	NO IMPORT - OPTION ABORT
TRND01	TRAN	IMS1	6D	INVALID PROGRAM NAME
TRND11	TRAN	IMS1	9	NO IMPORT - OPTION ABORT
TRND123	TRAN	IMS1	9	NO IMPORT - OPTION ABORT
TRND01	TRANDESC	IMS1	9	NO IMPORT - OPTION ABORT

34

RDDS Utility Screenshot Examples

RDDS Extraction Utility- Example Screenshot



36

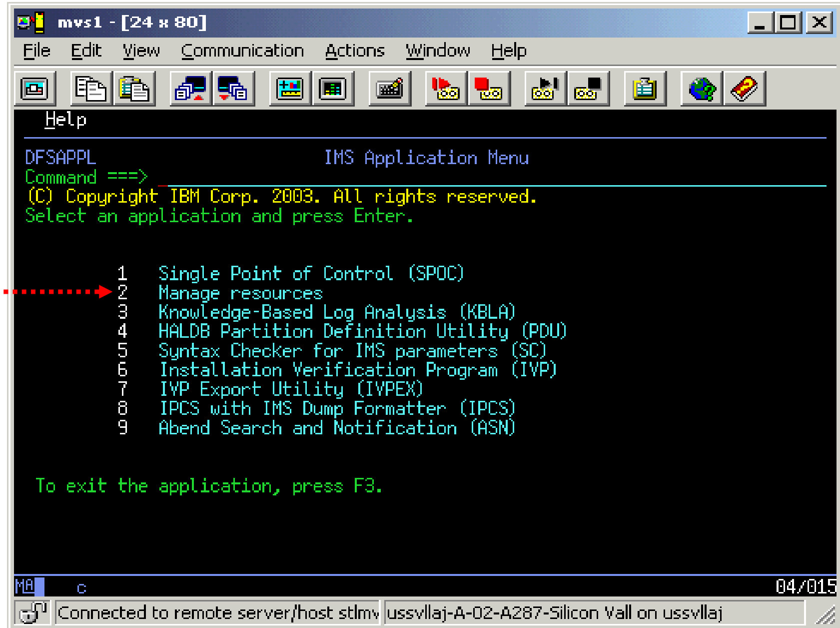
RDDS Extraction Utility – JCL Generated

```
//job name JOB  CLASS=J,MSGCLASS=A,MSGLEVEL=(1,1)
//JOBLIB   DD   DSN=[library data set name],DISP=SHR
//S1       EXEC PGM=DFSURDD0,MEMLIMIT=12G
//RDDSDSN  DD   DSN=[RDDS data set name],DISP=SHR
//SYSOUT   DD   DSN=[output data set name],DISP=(,CATLG,DELETE),
//              UNIT=SYSDA,VOL=SER=[Volume name],
//              SPACE=(CYL,(1,1),RLSE),
//              DCB=(LRECL=80,RECFM=FB,BLKSIZE=800)
//SYSPRINT DD   SYSOUT=*
//CONTROL    DD   *
OUTPUT=QUERY
/*
//
```

This example shows the JCL for querying the RDDS contents

New DRD Utilities - Example Screenshots

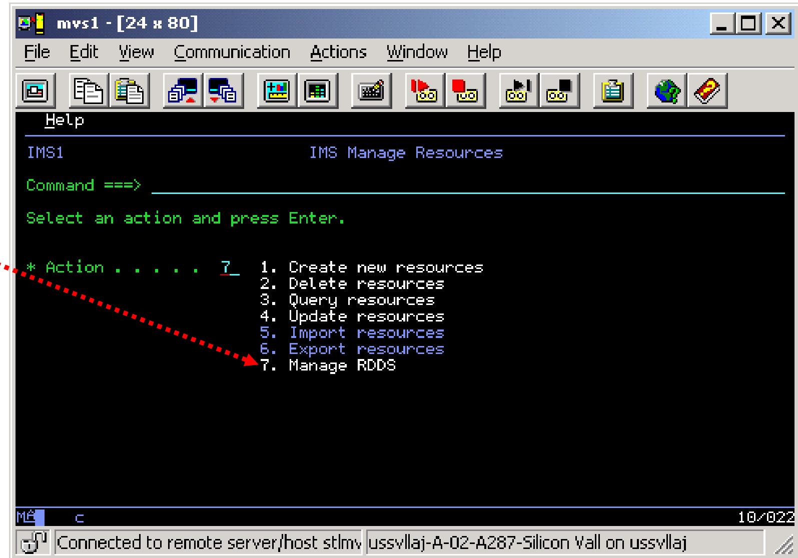
Invoke DRD Utilities via Manage Resource application



38

New DRD Utilities - Example Screenshots

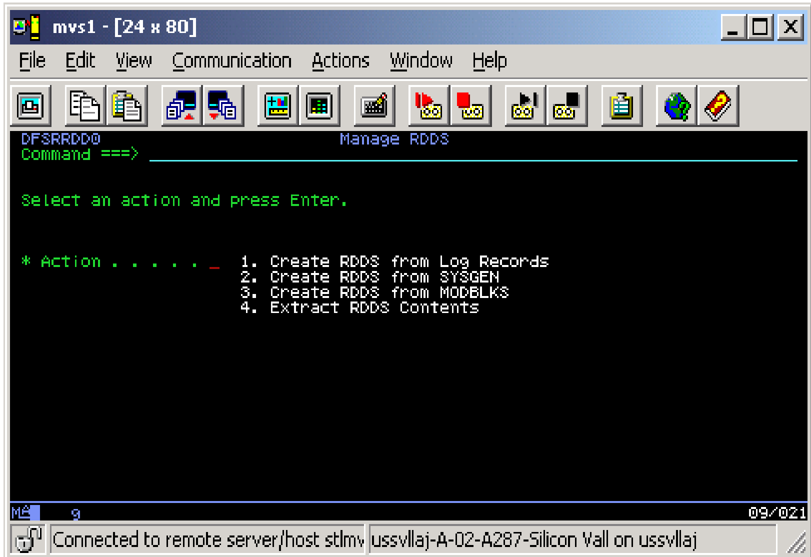
Select
Manage
RDDS



39

New DRD Utilities - Example Screenshots

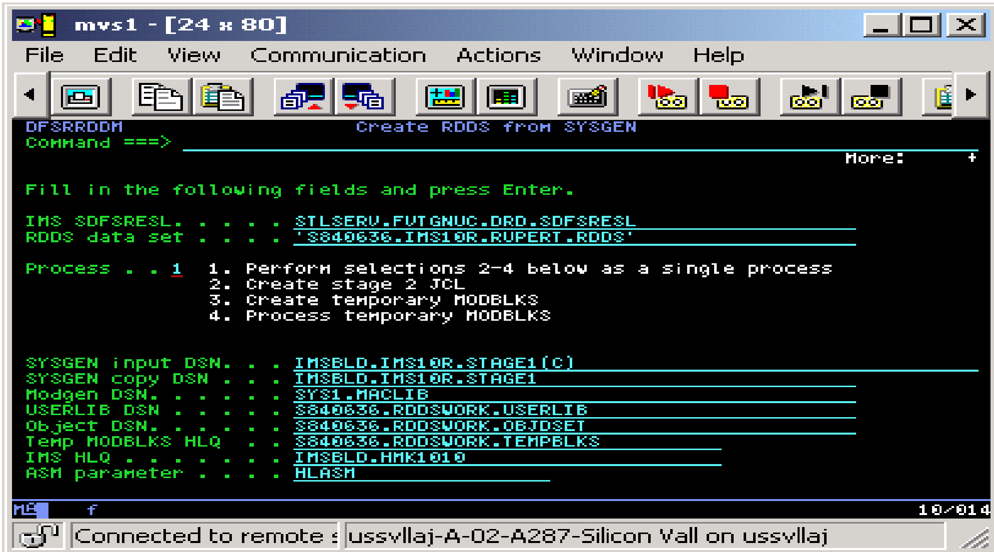
Select action to perform



40

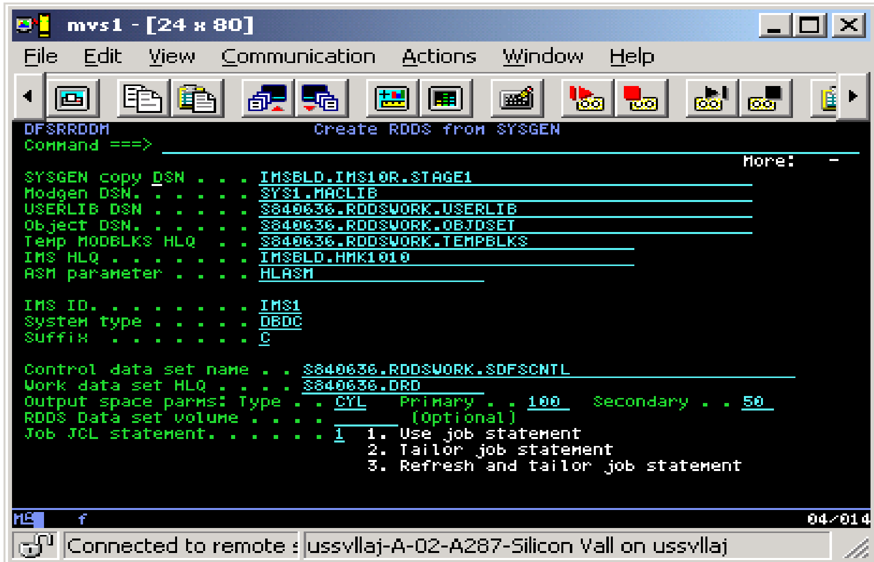
The new DRD utilities also are able to generate CREATE commands from both a MODBLKS data set and from Sysgen macros. However, they are not listed here on this “Manage RDDS” menu. The reason why is that to generate CREATE commands this way, an RDDS first needs to be created and subsequently have its contents extracted with the option of generating Type-2 commands.

Generating an RDDS from Sysgen – Panel 1



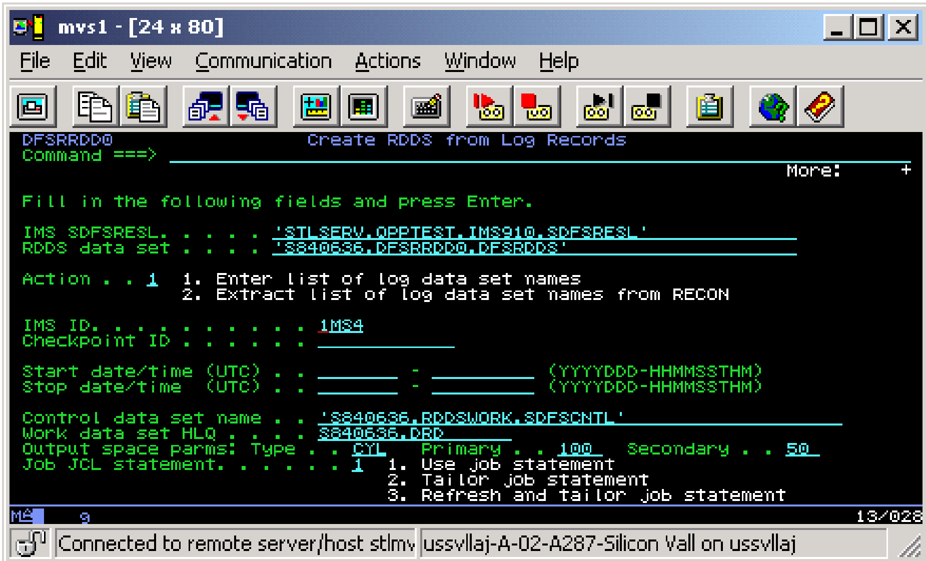
41

Generating an RDDS from Sysgen – Panel 2



42

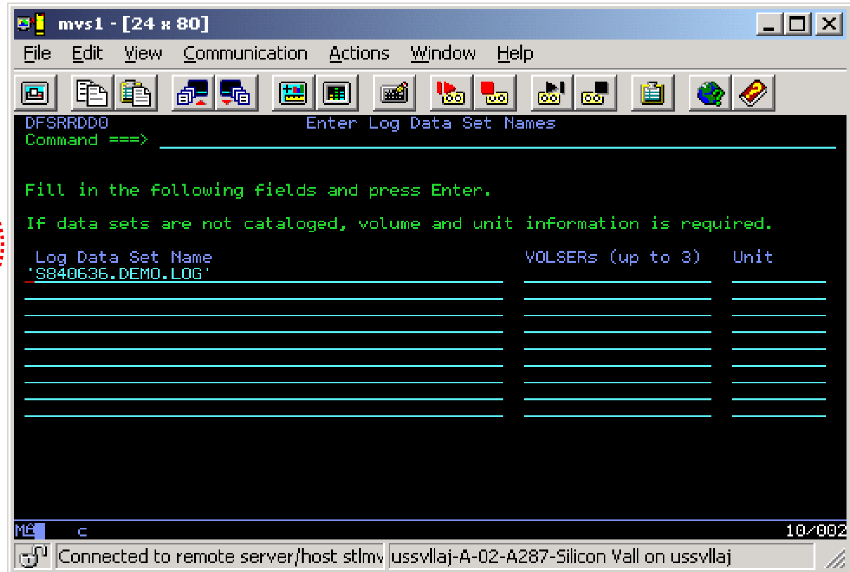
Generating an RDDS from X’22’ log records



43

Generating an RDDS from X'22' log records

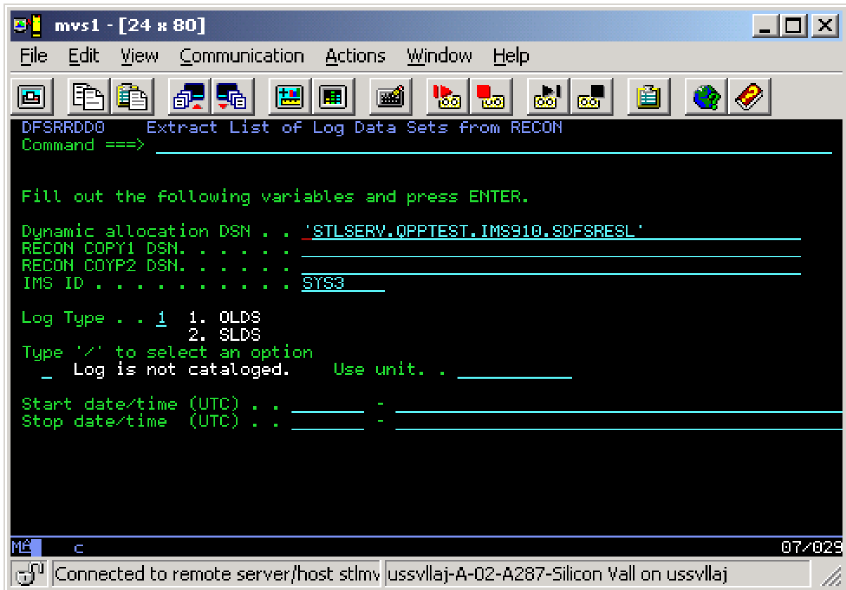
Option 1: enter log data set names



44

Generating an RDDS from X’22’ log records

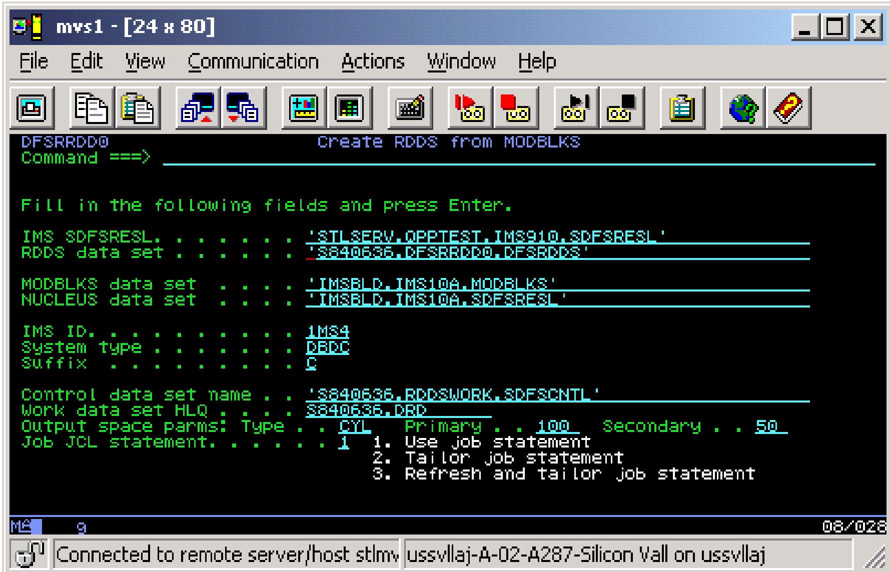
Option 2: enter RECON data set name



Generating an RDDS from X'22' log records – JCL generated from panels

```
//job name JOB  CLASS=J,MSGCLASS=A,MSGLEVEL=(1,1)
//JOBLIB   DD   DSN=[library data set name],DISP=SHR
//S1       EXEC PGM=DFSURCL0
//SYSUT1   DD   DSN=[Log data set name(s)],DISP=SHR
//RDDSDSN  DD   DSN=[RDDS data set name],DISP=(,CATLG,DELETE),
//              UNIT=SYSDA,VOL=SER=[Volume name],
//              SPACE=(CYL,(1,1),RLSE),
//              DCB=(LRECL=32756,BLKSIZE=32760,RECFM=VB)
//WORKFILE DD   DSN=[Workfile data set name),DISP=(,CATLG,DELETE),
//              UNIT=SYSDA,VOL=SER=[Volume name],
//              SPACE=(CYL,(1,1),RLSE),
//              DCB=(LRECL=133,BLKSIZE=6118,RECFM=FBA)
//REPORT   DD   SYSOUT=*,
//              DCB=(LRECL=133,BLKSIZE=6118,RECFM=FBA)

//CONTROL  DD   *
IMSID=SYS3
CHKPTID=200809002144935
/*
//
```

Generating an RDDS from MODBLKS



47

Generating an RDDS from MODBLKS – JCL generated from panels

```
job name JOB  CLASS=J,MSGCLASS=A,MSGLEVEL=(1,1)
//JOBLIB   DD   DSN=[library data set name],DISP=SHR
//S1       EXEC PGM=DFSURCM0
//MODBLKS  DD   DSN=[MODBLKS data set name],DISP=SHR
//RDDSDSN  DD   DSN=[RDDS data set name],DISP=(,CATLG,DELETE),
//              UNIT=SYSDA,VOL=SER=[Volume name],
//              SPACE=(CYL,(1,1),RLSE),
//              DCB=(LRECL=32756,BLKSIZE=32760,RECFM=VB)
//SYSPRINT DD   SYSOUT=*,
//              DCB=(LRECL=133,BLKSIZE=6118,RECFM=FBA)
//REPORT   DD   SYSOUT=*,
//              DCB=(LRECL=133,BLKSIZE=6118,RECFM=FBA)
//CONTROL  DD   *
IMSID=SYS3
SUF=V
/*
```

ACBLIB Usability Enhancements

This topic discusses two ACBLIB usability enhancements in IMS 11.

ACBLIB Usability Enhancements

- Dynamic allocation of ACBLIB data sets
- 64-bit ACB storage pool

There are two enhancements for ACBLIB usability in IMS 11. They are the capability to dynamically allocate the ACBLIB data sets and a new 64-bit storage pool that contains ACBs.

Dynamic allocation of ACBLIB data sets

- IMS outage required prior to IMS 11 if you need to resize the ACBLIB data sets since they are allocated via JCL
- In IMS 11, DFSMDA can optionally be used instead of JCL allocation
  - Inactive ACBLIB dynamically allocated only when needed
  - Active ACBLIB will be allocated all the time
- Now the inactive ACBLIB can be resized when necessary
  - Perform an online change to switch the newly resized inactive ACBLIB to the active ACBLIB to provide more space for the active ACBLIB
  - Resize the new inactive ACBLIB (previously too small active ACBLIB)
- Benefits
  - Manageability of ACBLIB is now improved
  - Users can increase the size of ACBLIB data sets without an outage

51

Prior to IMS 11 and this new capability, if you need to resize the ACBLIB data sets, you must take down IMS to do that because they are allocated via JCL, causing unavailability to end users. IMS 11 provides the optional capability to use DFSMDA dynamic allocation rather than JCL to allocate the ACBLIB data sets. Though dynamic allocation will be used for the active ACBLIB, it will be allocated all the time. However, the inactive ACBLIB will only be allocated when needed; therefore, since the inactive will be deallocated most of the time, it can be resized. Then a subsequent online change can be used to switch the resized inactive ACBLIB to be the active ACBLIB and the resizing will take effect. The now inactive ACBLIB (previously active ACBLIB) can then be resized to match the new active ACBLIB.

This capability improves the manageability of the ACBLIB data sets.

.

Dynamic allocation of ACBLIB data sets

- At control region initialization, IMS checks for the presence of the IMSACBA / IMSACBB DD statements
  - If they exist, no DFSMDA members are used (same as pre-IMS 11)
  - If they do not exist, DFSMDA members are used

52

IMS control region initialization will check to see if IMSACBA and IMSACBB DD statements exist; if so, no dynamic allocation will be done for the ACBLIB data sets; if not, IMS will dynamically allocate the ACBLIB data sets using DFSMDA members.

Dynamic allocation of ACBLIB data sets

- When DFSMDA members are used to define the ACBLIB data sets
  - Need to create a DFSMDA member for each of the IMSACBA and IMSACBB data set concatenations

```
DFSMDA TYPE=INITIAL
DFSMDA TYPE=IMSACBA
DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB1
DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB2
DFSMDA TYPE=FINAL
```

```
DFSMDA TYPE=INITIAL
DFSMDA TYPE=IMSACBB
DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB3
DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB4
DFSMDA TYPE=FINAL
```

  - Must remove JCL DD statements from the IMS procedure and the DLISAS procedure

53

Here are examples of using the DFSMDA macro to create members that will be used for dynamically allocating the ACBLIBA and ACBLIBB data sets. There are two new DFSMDA TYPEs, DFSMDA TYPE=IMSACBA and DFSMDA TYPE=IMSACBB, that will be used for dynamically allocating the two ACBLIB data sets. These examples show having two concatenations for each of the ACBLIB data sets.

The DFSMDA members are stored in one of the STEPLIB concatenations of the IMS procedure. IMSDALIB is also supported

Dynamic allocation of ACBLIB data sets

- Only the active ACBLIB datasets are allocated at control region initialization
- The inactive ACBLIB data sets are dynamically allocated during an online change process
- After an online change, the inactive ACBLIB data sets are deallocated
- The same DFSMDA member is used to allocate data sets in both CTL and DLISAS
  - No inconsistencies between CTL and DLISAS can exist
- Dynamic allocation of ACBLIB data sets is supported in all online configurations (IMS/TM, DBCTL, DCCTL, SAS and non-SAS, XRF, FDBR)
  - Not supported in batch

54

With DFSMDA, the active ACBLIB data set is dynamically allocated at IMS initialization and remains allocated for the duration of the IMS subsystem. With DFSMDA, the inactive ACBLIB is only allocated when needed to perform an online change. Once the online change is complete, the inactive ACBLIB data set will be deallocated.

Without dynamic allocation of ACBLIB data sets, both CTL and DLISAS need DD statements for the ACBLIBs so there can be inconsistencies between these two sets of DD statements. When DFSMDA is used, these inconsistencies cannot occur.

Dynamic allocation of ACBLIB data sets is not supported in batch. It is supported in all online configurations (IMS/TM, DBCTL, DCCTL, SAS and non-SAS, XRF, FDBR).

Dynamic allocation of ACBLIB data sets

- Can be used for correcting errors with an unusable inactive ACBLIB
  - Due to error in copying staging ACBLIB to inactive ACBLIB in preparation for an online change
- Additional data sets can be added or changes made to the current data sets in the inactive ACBLIB concatenation

55

There are other situations where dynamic allocation of ACBLIB is helpful.

If a problem is encountered in copying from the staging ACBLIB to the inactive ACBLIB in preparation for an online change, the inactive ACBLIB becomes unusable. Prior to IMS 11 an outage is needed to make it usable again. In IMS 11, it can be deleted and reallocated to make it usable without an outage.

Since the DFSMDA members describe all the concatenations for the inactive ACBLIB, additional concatenations can be added or changes made to existing concatenations for the inactive ACBLIB. Then an online change can apply these concatenation additions/changes to the active ACBLIB.

Dynamic allocation of ACBLIB data sets

- /DISPLAY MODIFY ALL command output now indicates new status for IMSACBA and IMSACBB
  - (A) Active, (I) Inactive, (U) Unallocated, ( ) DFSMDA not used

```
     LIBRARY  IMSACBA  (A)  IMSTESTG.DELTA1
                       (A)  IMSTESTG.IMS10AC.ACBLIB1
                       (A)  IMSTESTG.IMS10A.ACBLIB1
     LIBRARY  FORMATA  (A)  IMSTESTG.MFS.FORMAT1
                       (A)  IMSTESTG.MFS.FORMAT2
                       (A)  IMSQA.FMT1
     LIBRARY  MODBLKSA (A)  IMSBLD.I10ATS17.COMBLKS1
     LIBRARY  IMSACBB  (U)  IMSTESTG.DELTA2
                       (U)  IMSTESTG.IMS10AC.ACBLIB2
                       (U)  IMSTESTG.IMS10A.ACBLIB2
OR   LIBRARY  IMSACBB  ( )  NO DFSMDA MEMBER
     LIBRARY  FORMATB  (I)  IMSTESTG.MFS.FORMAT3
                       (I)  IMSTESTG.MFS.FORMAT4
                       (I)  IMSQA.FMT1
     LIBRARY  MODBLKSB (I)  IMSBLD.I10ATS17.COMBLKS2
     DISPLAY MODIFY COMPLETE *08230/110121*   SYS3
```

56

The /DISPLAY MODIFY ALL command output has been enhanced for the inactive ACBLIB to show a new status ‘U’ meaning unallocated when using DFSMDA or ( ) when there is neither a DD statement not a DFSMDA member for the library.

Dynamic allocation of ACBLIB data sets

- Benefits
  - Allows for resizing of online ACBLIB data sets without an outage
  - Allows for correcting errors with the inactive ACBLIB
  - Allows for adding additional data seta to the ACBLIB concatenation without an outage
  - Provides ACBLIB consistency between CTL and DLISAS
  - Improves online availability
    - Prior to IMS 11 the required outage is typically a ‘planned’ outage
    - With IMS 11 no outage is necessary to resize the ACBLIB data sets

57

The benefits of the IMS 11 function for dynamically allocating ACBLIB data sets improve IMS online system availability by removing the outage that is necessary today to handle problems with ACBLIB data set sizing, to handle problems with the inactive ACBLIB, and to allow for changes in the ACBLIB concatenations.

An outage, whether planned or unplanned, is no longer required in IMS 11.

64-bit ACB storage pool

- An optional 64-bit storage pool to cache ACB members can be created in IMS 11 for non-resident PSBs and DMBs
- The goal is to improve storage utilization and performance
  - Reduces I/Os to the ACBLIB
  - Improves ACBLIB performance for customers with large ACBLIBs
  - May be used to reduce the size of the PSB pools without causing additional I/Os.

58

The 64-bit ACB storage pool enhancement provides a separate pool for non-resident ACBs as an option to improve storage utilization and performance for those customers who have heavy I/O activity to the ACBLIB or have large ACBLIBs with many members.

This enhancement may also be used to reduce the size of the PSB pools without causing additional I/Os. When a PSB pool is very large with many members in the pool, IMS can use many instructions to manage the pool when members have to be removed from the pool to make room for another member. These instructions are used to find the “best” member(s) to cast out of the pool. In these cases IMS uses significantly less CPU when the pool is smaller but the members to be added to the PSB pools are already in the 64-bit ACB storage pool.

Processing for ACB members

- Prior to IMS 11
  - At control region initialization
    - Resident DMBs and PSBs are loaded into 31-bit extended storage of DLISAS or CTL (w/o DLISAS)
    - DEDBs are loaded into ECSA
  - During execution
    - Non-resident DMBs and PSBs are loaded on demand into 31-bit non-resident pools (DMB and PSB pools)

59

Prior to IMS 11, ACBs are handled as described above. At control region initialization, resident PSBs and resident DMBs as well as DEDBs are loaded. Then during execution non-resident PSBs and DMBs are loaded as needed into the 31-bit non-resident pools (DMB and PSB pools).

Processing of ACB members in IMS 11

- With IMS 11 and 64-bit ACB storage pool
  - At control region initialization
    - Same as pre-IMS 11
  - During execution
    - Non-resident DMBs and PSBs are loaded into 64-bit ACB storage pool after being loaded on demand into 31-bit non-resident pools
    - Resident DMBs and PSBs will not go into 64-bit ACB storage pool
    - DEDBs will not go into 64-bit ACB storage pool
- Supported in all online configurations (IMS/TM, DBCTL, DCCTL, SAS and non-SAS, XRF, FDBR)
    - DCCTL only has PSBs
    - No batch support
    - DOPT PSBs not supported

60

With IMS 11 and the presence of a 64-bit ACB storage pool, control region initialization remains the same (resident PSBs, resident DMBs, and DEDBs are loaded). However, at execution time, as non-resident PSBs and DMBs are loaded from ACBLIB into the 31-bit non-resident pools, these non-resident PSBs and DMBs are also loaded into the 64-bit ACB storage pool, so they will be more easily accessible later.

All online configurations of IMS have support for a 64-bit ACB storage pool. There is no support for this capability in batch. DOPT PSBs are not supported.

Specifying the 64-bit ACB storage pool

- Specification of the 64-bit ACB storage pool is in the new DATABASE section of the DFSDFxxx PROCLIB member
- Parameter is ACBIN64=nnn where nnn is the number of gigabytes for the 64-bit ACB storage pool (1-999)
- 64-bit ACB pool needs to be large enough to contain both non-resident PSBs and non-resident DMBs
  - Minimum would be sum of sizes of 31-bit non-resident PSB and DMB pools
  - Maximum would be total size of all non-resident ACB members
- Recommendation is to start with 1 or 2 gigabytes

```
<SECTION=DATABASE>
ACBIN64=1
```


The 64-bit ACB storage pool is defined in a new section of the DFSDFxxx PROCLIB member called DATABASE. The parameter that must be specified is ACBIN64=nnn where nnn is the number of gigabytes of storage for this new pool. If the ACBIN64 parameter is not present, the 64-bit ACB storage pool will not be created and used.

64-bit ACB storage pool considerations

- Scheduling considerations with non-resident ACB resources
  - At first scheduling of a program, a PSB and any related DMBs are loaded into the 31-bit non-resident pools and are also loaded into the 64-bit ACB storage pool
  - At subsequent schedulings of this program, ACB members not found in the 31-bit non-resident pools are copied from the 64-bit ACB storage pool to the 31-bit non-resident pools (which avoids I/Os to ACBLIB)
  - If the 64-bit ACB storage pool is full, a LRU algorithm will be used to remove old members to make room for new members

62

This chart describes how scheduling with non-resident ACB resources works with the new 64-bit ACB storage pool.

At the first scheduling of a program, its PSB and any related DMBs are loaded into the 31-bit non-resident pools and are also loaded into the 64-bit ACB storage pool. At subsequent schedulings of this program, ACB members not found in the 31-bit non-resident pools are copied from the 64-bit ACB storage pool to the 31-bit non-resident pools, avoiding an I/O to ACBLIB.

If the 64-bit ACB storage pool is full, a LRU algorithm will be used to remove old members to make room for new members.

64-bit ACB storage pool considerations

- Online change considerations
  - ACB members affected by the online change will be
    - Removed from the 31-bit non-resident pools
    - Deleted from the 64-bit ACB storage pool
- Type-2 DELETE command considerations
  - DELETE DB and DELETE PGM will
    - Remove ACB members from the 31-bit non-resident pools
    - Delete ACB members from the 64-bit ACB storage pool

63

During an online change, if a 64-bit ACB storage pool exists, ACB members affected by the online change will be removed from the 31-bit non-resident pools and deleted from the 64-bit ACB storage pool.

When using the type-2 DELETE DB or DELETE PGM commands and a 64-bit ACB storage pool exists, these commands will also remove ACB members from the 31-bit non-resident pools and delete ACB members from the 64-bit ACB storage pool.

64-bit ACB storage pool considerations

- Impact on managing DMBs
  - DMBs today are either defined as resident or are always in the non-resident DMB pool
  - Most likely new 64-bit ACB storage pool will have minimal impact
- Impact of managing PSBs
  - Will depend on scheduling patterns (WFIs have no scheduling)
  - PSBs defined as resident today
    - If large number of PSBs to be scheduled, investigate reducing/eliminating resident PSBs, increasing the size of the non-resident PSB pool, and using the 64-bit ACB storage pool
      - No noticeable performance impact of retrieving the PSB from 64-bit ACB pool versus from the resident PSB pool
  - PSBs using the non-resident PSB pool today
    - If non-resident PSB pool is sized larger to reduce/eliminate ACBLIB I/Os, investigate using a smaller non-resident PSB pool with the 64-bit ACB storage pool
    - 64-bit ACB pool removes potential I/O for PSBs

64

Most likely the new 64-bit ACB storage pool will have minimal impact on DMB manageability because today they are either resident or for non-resident DMBs, the non-resident DMB pool is sized large enough to hold all non-resident DMBs.

PSB manageability may have some different considerations with the 64-bit ACB storage pool, though actual impact will be based on the scheduling patterns of each IMS system. For PSBs that are defined as resident today, if many of these are scheduled, it may be more efficient to make some or all of them non-resident, increase the size of the non-resident PSB pool, and use the 64-bit ACB storage pool to access them when needed. You can using the saving from reducing/eliminating resident PSBs to increase the size of the non-resident PSB pool. For PSBs that are non-resident today, if the non-resident PSB pool has been sized larger to reduce/eliminate ACBLIB I/Os, then using the 64-bit ACB storage pool would make it possible to reduce the size of the non-resident PSB pool. This would be helpful for customers with large numbers of PSBs that cannot be always found in the non-resident PSB pool today.

IBM

Querying the 64-bit ACB storage pool

- A new QUERY POOL TYPE(ACBIN64) can be used to monitor the usage of the 64-bit pool

```
QUERY POOL TYPE(ACBIN64)
```

PoolName	Type	CC	Size	Mbrs	Used
ACBIN64	Cache64	0	4096	10000	60

```
QUERY POOL TYPE(ACBIN64) SHOW(ALL)
```

PoolName	Type	CC	Size	Mbrs	Used	Free	Overflow	Gets	Hit	Miss
ACBIN64	Cache64	0	1024	3700	25	3675	0	1000	900	100

Isrt	Del	Lmbr	Lsize	Smbr	Ssize
300	20	PAYROLL	2000	DEBIT	100

Here is an example of the new formats for QUERY POOL TYPE(ACBIN64) command.

Statistics for the 64-bit ACB storage pool

- New log record – type X'4515'
  - Contains statistics from new QUERY POOL TYPE(ACBIN64) command

PoolNm	-	Pool name (ACBIN64)
Type	-	CACHE64
Size	-	Pool size in megabytes
Mbrs	-	Total number of buffers stored in the pool, whether in use or not
Used	-	Number of buffers currently in use (number of ACBLIB members in pool)
Free	-	Total number of buffers allocated in the pool but not in use
Overflow	-	Total number of overflow buffers in use
Gets	-	Number of FIND calls, whether successful or not
Hit	-	Number of successful FIND calls
Miss	-	Number of unsuccessful FIND calls
Isrt	-	Number of buffers added to the pool
Del	-	Number of buffers deleted from the pool, including castouts
Lmbr	-	Name of largest member in the 64-bit pool.
Lsize	-	Size in kilobytes (K) of the largest member
Smbr	-	Name of smallest member in the 64-bit pool.
Ssize	-	Size in kilobytes (K) of the smallest member.

66

The same type of information that is provided by a type-2 QUERY POOL TYPE(ACBIN64) command will be logged in a new type '4515' checkpoint log record.

Monitoring for the 64-bit ACB storage pool

- New monitor record types
  - Type 74
    - Issued when a get request for a PSB in the 64-bit pool is started
  - Type 75
    - Issued when a get request for a PSB in the 64-bit pool ends
  - Type 76
    - Issued when a get request for a DMB in the 64-bit pool is started
  - Type 77
    - Issued when a get request for a DMB in the 64-bit pool ends
  - Supported by IMS Monitor in the Region IWAIT Scheduling + Termination report
  - Supported by IMS PA tool

The IMS Monitor will record four new record types for usage of the 64-bit ACB storage pool: type 74 indicates that a get request for a PSB in the 64-bit pool has started, type 75 indicates that a get request for a PSB in the 64-bit pool has ended, type 76 indicates that a get request for a DMB in the 64-bit pool has started, and type 77 indicates that a get request for a DMB in the 64-bit pool has ended.

These new monitor record types are supported by the IMS Monitor and the IMS PA tool.

64-bit ACB storage pool – IMS Monitor REGION IWAIT report

```
 IMS MONITOR   *** REGION IWAIT ***               TRACE START 2010 123, 08:01:32  TRACE STOP 2010 123, 08:11:48

.........IWAIT TIME..........
**REGION     2  OCCURRENCES       TOTAL        MEAN     MAXIMUM       FUNCTION  MODULE
  ______     ___________       _____        ____     _______       ________  ______
SCHEDULING + TERMINATION___   ___________
                         2    32975611    16487805    23293621    NO MESSAGES   MSC
...SUB-TOTAL...
   _________             2    32975611    16487805
                         1          39          39          39    PSB=DDLTRN24  BLR-64BIT
                         1        1965        1965        1965    PSB=BMPFPE07  BLR
                         1        1013        1013        1013    PSB=BMPFPE05  BLR
                         1          38          38          38    PSB=BMPFPE02  BLR-64BIT
                         2          41          35          35    PSB=DRP255    BLR-64BIT
..TOTAL...
   _____                 8    32978707     4122325
DL/I CALLS
____ _____
```

68

This is an example of the IMS Monitor REGION IWAIT report that shows activity in the 64-bit ACB storage pool.

64-bit ACB storage pool summary

- Benefits
  - Improved technique for better management of non-resident ACBs
  - Goal is to improve storage utilization and performance for ACBs
    - Reduces I/Os to the ACBLIB
    - Improves ACBLIB performance for customers with large ACBLIBs
  - Improves ACB usability for customers where ACBLIB access impacts performance and growth

IMS 11 provides better management for non-resident ACBs by providing a 64-bit caching capability that will improve storage utilization and performance for ACBLIB operations. Using this function will reduce I/Os to the ACBLIB data set and improve performance for customers with large ACBLIBs.

V10 SPOC Print Options

This topic discusses new printing options for the SPOC in IMS 10 via PTF. This support is included in IMS 11.

SPOC Print Options Support

- New print options support for a SPOC when running in a CSL environment with OM
  - Available in IMS 10 via APAR PK50292 / PTF UK35086 (May 2008)
  - Included in IMS 11
- Applies to
  - TSO SPOC
  - batch SPOC (IMS 10)
  - REXX SPOC API
  - OM audit trail CSLULALE (formatted) print exit with DFSERA10 (IMS 10)
- Focus is improving readability / usability of printed output from a SPOC

71

New print formatting options that address enhanced usability for printed output from a SPOC are available in IMS 10 as APAR PK50292 / PTF UK35086 which was available in May 2008. This support is included in IMS 11.

This support applies to all types of SPOC – the TSO SPOC, the batch SPOC, the REXX SPOC API – and also to the use of the CSLULALE print exit with DFSERA10 for printing the OM audit trail.

SPOC Print Options Support

- 3 styles of formatting
  - WRAP (default)
    - Lines of data are fit to the width of the print file and wrapped to the next line as needed
    - Works this way before enhancement – very hard to read
  - BYCOL
    - Lines of data are grouped together by column
    - Example: all records that include columns 3-8 are printed followed by all records that include columns 9-12
  - BYRSC
    - Lines of data are grouped together by resource
    - Example: all records that include resources 1-7 are printed followed by all records that include resources 8-15

72

There are three styles of formatting that can be selected when printing from a SPOC. ‘WRAP’ is always the default.

Using SPOC Print Options Support

- From TSO SPOC
  - Options -> Preferences
    - Format of listing
      - 1. Wrap individual lines.
      - 2. Group lines by column.
      - 3. Group lines by resource.
  - Print via action bar
    - File -> Print or File -> Print all
    - File -> Save as
- For batch SPOC
  - EXEC PGM=CSLUSPOC,
    PARM=('IMSPLEX=PLEX1,..F=WRAP|BYCOL|BYRSC')

73

This chart shows how to implement these new print options for the TSO SPOC and for the batch SPOC.

Using SPOC Print Options Support

- For REXX SPOC API
  - rexx function CSLULOPT enhanced to support two additional parameters
    - x = CSLULOPT('LRECL=number')
      - number = numeric value of the logical record length (default 133)
    - x = CSLULOPT('F=opt')
      - opt = WRAP - wrap individual lines
      - BYCOL - group lines by column
      - BYRSC - group lines by resource
  - rexx function CSLULGTP enhanced to utilize the specified print options when creating the .report stem variable
- For OM audit trail log print using DFSERA10
  - OPTION PRINT, EXITR=CSLULALE,PARM=(F=WRAP|BYCOL|BYRSC)

74

This chart shows how to implement these new print options for the REXX SPOC API and OM audit trail log print.

Using SPOC Print Options Support
Wrap example: 1 heading line / 1 data line

```
Trancode MbrName    CC PSBname  LCls    LQCnt  LLCT LPLCT LPLCTTime LCPRI LNPRI LLPRI LSegSz LSegNo LParLim RegCnt
         LMaxRgn EditRtn  FP EMHBSz CmtMode MsgType SPATrunc SPASz SIDR SIDL DCLWA DirRoute EditUC Inq Recover Resp
         Remote Serial WFI AOCMD Conv TranStat LclStat ModelName ModelType MSName   TimeAccess
         TimeUpdate           TimeCreate           TimeImport           DefnType
-------------------------------------------------------------------------------------------------------------------
ADDINV   IMS1        0 DFSSAM04    4        0     2 65535   6553500     7     7    10      0      0   65535      0
               0          N       0 MULT    MULTSEG              0   10   10 Y     N        Y      N   Y       N
         N      N      N   N     N    N
                              2008.267 12:32:05.43                      MODBLKS
ADDPART  IMS1        0 DFSSAM04    4        0     2 65535   6553500     7     7    10      0      0   65535      0
               0          N       0 MULT    MULTSEG              0   10   10 Y     N        Y      N   Y       N
         N      N      N   N     N    N
                              2008.267 12:32:05.43                      MODBLKS
AOBMP    IMS1        0 TS2IAOB0   23        0 65535 65535   6553500     0     0     0      0      0   65535      0
               0          N       0 SNGL    MULTSEG              0   10   10 Y     N        Y      N   Y       N
         N      N      Y   N     N    N
                              2008.267 12:32:05.43                      MODBLKS
AOP      IMS1        0 TS1IAOP0    4        0     4     4       500    10    10    12      0    500   65535      0
               0          N       0 SNGL    SNGLSEG              0   10   10 Y     N        Y      N   Y       Y
         N      N      N   N     N    N
                              2008.267 12:32:05.43                      MODBLKS
APOL11   IMS1        0 APOL1       1        0 65535 65535   6553500     1     1     1      0      0   65535      0
               0          N       0 MULT    MULTSEG              0   10   10 Y     N        Y      Y   Y       Y
         N      N      N   N     N    N
                              2008.267 12:32:05.43                      MODBLKS
APOL12   IMS1        0 APOL1       1        0 65535 65535   6553500     9     9     9      0      0   65535      0
               0          N       0 MULT    MULTSEG              0   10   10 Y     N        Y      N   Y       N
```


This is an example of the ’WRAP’ option printed output (the default) for the results of a type-2 QUERY TRAN NAME(*) SHOW(ALL) command.

The heading line is wrapped across multiple print output lines, making it hard to use the heading columns with the resource lines below it.

Each resource line contains all of the statuses/attributes before another resource is printed.

This is the way SPOC printed output has looked before this new enhancement.

IMS Version 11

Using SPOC Print Options Support By Column Example (Page 1)

Trancode	MbrName	CC	PSBname	LCls	LQCnt	LLCT	LPLCT	LPLCTTime	LCPRI	LNPRI	LLPRI	LSegSz	LSegNo	LParLim	RegCnt
--------	--------	----	--------	----	--------	-----	-----	---------	-----	-----	-----	------	------	-------	------
ADDINV	IMS1	0	DFSSAM04	4	0	2	65535	6553500	7	7	10	0	0	65535	0
ADDPART	IMS1	0	DFSSAM04	4	0	2	65535	6553500	7	7	10	0	0	65535	0
AOBMP	IMS1	0	TS2IAOB0	23	0	65535	65535	6553500	0	0	0	0	0	65535	0
AOP	IMS1	0	TS1IAOP0	4	0	4	4	500	10	10	12	0	500	65535	0
APOL11	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
APOL12	IMS1	0	APOL1	1	0	65535	65535	6553500	9	9	9	0	0	65535	0
APOL13	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	5	3	65535	0
APOL14	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	65535	65535	65535	0
APOL15	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	80	1	65535	0
APOL16	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	80	3	65535	0
APOL17	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
APOL18	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
APOL21	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
APOL22	IMS1	0	APOL1	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
AUTRAN1H	IMS1	0	AUTPSB1H	1	0	2	65535	6553500	7	7	10	0	0	65535	0
AUTRAN11	IMS1	0	AUTPSB11	1	0	2	65535	6553500	7	7	10	0	0	65535	0
AUTRAN12	IMS1	0	AUTPSB11	1	0	2	65535	6553500	7	7	10	0	0	65535	0
AUTRAN2H	IMS1	0	AUTPSB1H	1	0	2	65535	6553500	7	7	10	0	0	65535	0
A1111111	IMS1	0	A11APP	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
A3270	IMS1	0	A3270	1	0	65535	1	6553500	8	8	8	0	0	65535	0
BACKORDR	IMS1	0	TPARTAPP	1	0	2	65535	6553500	7	7	10	0	0	65535	0

76

This is an example of the first page of the 'BYCOL' option printed output for the results of a type-2 QUERY TRAN NAME(*) SHOW(ALL) command.

All resources are shown with the same set of columns for as many pages as it takes to show all resources, followed by another set of pages with the next set of columns/etc. The new support determines how many columns can fit across one printed page. The 2 left-most columns (Trancode and MbrName) are repeated on the left of all pages.

This example shows 2 header columns followed by 14 data columns.

Using SPOC Print Options Support By Column Example (Page 2)

Trancode	MbrName	CC	PSBname	LCls	LQCnt	LLCT	LPLCT	LPLCTTime	LCPRI	LNPRI	LLPRI	LSegSz	LSegNo	LParLim	RegCnt
BHA1	IMS1	0	PMAPJK13	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHA2	IMS1	0	PMAPJK23	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHD1	IMS1	0	PMAPJK14	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHD2	IMS1	0	PMAPJK24	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHD3	IMS1	0	PMAPJK34	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHE1	IMS1	0	BMAPJK11	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHE2	IMS1	0	BMAPJK21	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHE4	IMS1	0	BMAPJK21	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHF1	IMS1	0	PMVAPZ12	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHF2	IMS1	0	PMVAPZ22	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHF3	IMS1	0	PMVAPZ32	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHF4	IMS1	0	PMVAAZ42	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHHA1	IMS1	0	PMHAJK13	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHHC1	IMS1	0	PMHCJK15	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHHD1	IMS1	0	PMHDJK14	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHHF1	IMS1	0	PMHFJK12	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHHX1	IMS1	0	PMHXJK19	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHI1	IMS1	0	PMAPJK15	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BHI2	IMS1	0	PMAPJK25	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BH2X0	IMS1	0	PM2XJK05	1	0	65535	65535	6553500	1	1	1	0	0	65535	0
BH2X1	IMS1	0	PM2XJK15	1	0	65535	65535	6553500	1	1	1	0	0	65535	0

77

This is an example of the second page of the 'BYCOL' option printed output for the results of a type-2 QUERY TRAN NAME(*) SHOW(ALL) command, showing the same columns as page 1 but for the next page of trancode resources. Note that Trancode and MbrName are repeated on the left. There will be as many following pages of this format as needed to show this set of columns for all trancode resources.

IMS Version 11

Using SPOC Print Options Support By Column Example (Page 6)

Trancode	MbrName	LMaxRgn	EditRtn	FP	EMHBSz	CmtMode	MsgType	SPATrunc	SPASz	SIDR	SIDL	DCLWA	DirRoute	EditUC	Inq
-------	--------	-------	--------	--	------	-------	-------	--------	-----	----	----	-----	--------	------	---
ADDINV	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	N
ADDPART	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	N
AOBMP	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
AOP	IMS1	0		N	0	SNGL	SNGLSEG		0	10	10	Y	N	Y	N
APOL11	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	Y
APOL12	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	N
APOL13	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	N
APOL14	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	Y
APOL15	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	N
APOL16	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	N
APOL17	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	Y
APOL18	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
APOL21	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	Y
APOL22	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	N
AUTRAN1H	IMS1	0		N	0	SNGL	SNGLSEG		0	10	10	Y	N	Y	N
AUTRAN11	IMS1	0		N	0	SNGL	SNGLSEG		0	10	10	Y	N	Y	N
AUTRAN12	IMS1	0		N	0	SNGL	SNGLSEG		0	10	10	Y	N	Y	N
AUTRAN2H	IMS1	0		N	0	SNGL	SNGLSEG		0	10	10	Y	N	Y	N
A1111111	IMS1	0		N	0	SNGL	SNGLSEG	S	80	10	10	Y	N	Y	N
A3270	IMS1	0		N	0	MULT	MULTSEG		0	10	10	Y	N	Y	N
BACKORDR	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N

78

After all the pages are printed for the first set of columns, then the next set of columns will be printed.

This is an example of a following page that shows another set of columns of the 'BYCOL' option printed output for the results of a type-2 QUERY TRAN NAME(*) SHOW(ALL) command.   This shows the two repeated columns on the left (Trancode, MbrName) followed by 13 columns of additional data.

Using SPOC Print Options Support By Column Example (Page 7)

Trancode	MbrName	LMaxRgn	EditRtn	FP	EMHBSz	CmtMode	MsgType	SPATrunc	SPASz	SIDR	SIDL	DCLWA	DirRoute	EditUC	Inq
BHA1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHA2	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHD1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHD2	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHD3	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHE1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHE2	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHE4	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHF1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHF2	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHF3	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	Y
BHF4	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	Y
BHHA1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHHC1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHHD1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHHF1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHHX1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHI1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BHI2	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BH2X0	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N
BH2X1	IMS1	0		N	0	SNGL	MULTSEG		0	10	10	Y	N	Y	N

This is an example of the second page of the second set of columns produced with the ’BYCOL’ option. There will be as many following pages of this format as needed to show this set of columns for all trancode resources.

Using SPOC Print Options Support By Column Example (Page 10)

Trancode	MbrName	Recover	Resp	Remote	Serial	WFI	AOCMD	Conv	TranStat	LclStat	ModelName	ModelType	MSName
ADDINV	IMS1	Y	N	N	N	N	N	N	N				
ADDPART	IMS1	Y	N	N	N	N	N	N	N				
AOBMP	IMS1	Y	N	N	N	Y	N	N	N				
AOP	IMS1	Y	Y	N	N	N	N	N	N				
APOL11	IMS1	Y	Y	N	N	N	N	N	N				
APOL12	IMS1	Y	N	N	N	N	N	N	N				
APOL13	IMS1	Y	N	N	N	N	N	N	N				
APOL14	IMS1	N	N	N	N	N	N	N	N				
APOL15	IMS1	Y	N	N	N	N	N	N	N				
APOL16	IMS1	Y	N	N	N	N	N	N	N				
APOL17	IMS1	Y	N	N	N	N	N	N	N				
APOL18	IMS1	Y	N	N	N	Y	N	N	N				
APOL21	IMS1	Y	Y	N	N	N	N	N	N				
APOL22	IMS1	Y	N	N	N	N	N	N	N				
AUTRAN1H	IMS1	Y	N	N	N	N	N	N	N				
AUTRAN11	IMS1	Y	N	N	N	N	N	N	N				
AUTRAN12	IMS1	Y	N	N	N	N	N	N	N				
AUTRAN2H	IMS1	Y	N	N	N	N	N	N	N				
A1111111	IMS1	Y	N	N	N	N	N	Y	N				
A3270	IMS1	Y	N	N	N	N	N	N	N				
BACKORDR	IMS1	Y	N	N	N	N	N	N	N				

80

After all the pages are printed for the second set of columns, then the next set of columns will be printed.

This is an example of a following page that shows another set of columns of the ’BYCOL’ option printed output for the results of a type-2 QUERY TRAN NAME(*) SHOW(ALL) command. This shows the two repeated columns on the left (Trancode, MbrName) followed by 12 columns of data. There will be as many following pages of this format as needed to show this set of columns for all trancode resources.

Following this set of columns for transactions will be the last set of pages showing the remaining columns from the QUERY output.

Using SPOC Print Options Support By Resource Example (Page 1)

```
Trancode MbrName    CC PSBname  LCls    LQCnt  LLCT LPLCT LPLCTTime LCPRI LNPRI LLPRI LSegSz LSegNo LParLim RegCnt
------- -------- ---- -------- ---- -------- ----- ----- --------- ----- ----- ----- ------ ------ ------- ------
ADDINV   IMS1        0 DFSSAM04    4        0      2 65535   6553500     7     7    10      0      0   65535      0
ADDPART  IMS1        0 DFSSAM04    4        0      2 65535   6553500     7     7    10      0      0   65535      0
AOBMP    IMS1        0 TS2IAOB0   23        0 65535 65535   6553500     0     0     0      0      0   65535      0
AOP      IMS1        0 TS1IAOP0    4        0      4     4       500    10    10    12      0    500   65535      0
Trancode MbrName  LMaxRgn EditRtn  FP EMHBSz CmtMode MsgType SPATrunc SPASz SIDR SIDL DCLWA DirRoute EditUC Inq
-------- -------- ------- -------- -- ------ ------- ------- -------- ----- ---- ---- ----- -------- ------ ---
ADDINV   IMS1           0          N       0 MULT    MULTSEG             0   10    10 Y     N        Y      N
ADDPART  IMS1           0          N       0 MULT    MULTSEG             0   10    10 Y     N        Y      N
AOBMP    IMS1           0          N       0 SNGL    MULTSEG             0   10    10 Y     N        Y      N
AOP      IMS1           0          N       0 SNGL    SNGLSEG             0   10    10 Y     N        Y      N
Trancode MbrName  Recover Resp Remote Serial WFI AOCMD Conv TranStat LclStat ModelName ModelType MSName
-------- -------- ------- ---- ------ ------ --- ----- ---- -------- ------- --------- --------- --------
ADDINV   IMS1     Y       N    N      N      N   N     N    N
ADDPART  IMS1     Y       N    N      N      N   N     N    N
AOBMP    IMS1     Y       N    N      N      Y   N     N    N
AOP      IMS1     Y       Y    N      N      N   N     N    N
Trancode MbrName  TimeAccess          TimeUpdate          TimeCreate          TimeImport          DefnType
-------- -------- ------------------- ------------------- ------------------- ------------------- --------
ADDINV   IMS1                                             2008.267 12:32:05.43                    MODBLKS
ADDPART  IMS1                                             2008.267 12:32:05.43                    MODBLKS
AOBMP    IMS1                                             2008.267 12:32:05.43                    MODBLKS
AOP      IMS1                                             2008.267 12:32:05.43                    MODBLKS
```

81

This is an example of the 'BYRSC' option printed output for the results of a type-2 QUERY TRAN NAME(*) SHOW(ALL) command.

Information is grouped by resource. All columns for particular resources are shown together to fit on one page. The new support will determine how many resources can fit on one page.

This example shows printing all the information for 4 transactions in a way that is more readable than the WRAP method. Notice that the two columns, Trancode and MbrName, are repeated for each set of columns.

Using SPOC Print Options Support By Resource Example (Page 2)

```
Trancode MbrName    CC PSBname  LCls    LQCnt  LLCT LPLCT LPLCTTime LCPRI LNPRI LLPRI LSegSz LSegNo LParLim RegCnt
-------- -------- ---- -------- ---- -------- ----- ----- --------- ----- ----- ----- ------ ------ ------- ------
APOL11   IMS1        0 APOL1       1        0 65535 65535   6553500     1     1     1      0      0   65535      0
APOL12   IMS1        0 APOL1       1        0 65535 65535   6553500     9     9     9      0      0   65535      0
APOL13   IMS1        0 APOL1       1        0 65535 65535   6553500     1     1     1      5      3   65535      0
APOL14   IMS1        0 APOL1       1        0 65535 65535   6553500     1     1     1  65535  65535   65535      0
Trancode MbrName  LMaxRgn EditRtn  FP EMHBSz CmtMode MsgType SPATrunc SPASz SIDR SIDL DCLWA DirRoute EditUC Inq
-------- -------- ------- -------- -- ------ ------- ------- -------- ----- ---- ---- ----- -------- ------ ---
APOL11   IMS1           0          N       0 MULT    MULTSEG              0   10   10 Y     N        Y      Y
APOL12   IMS1           0          N       0 MULT    MULTSEG              0   10   10 Y     N        Y      N
APOL13   IMS1           0          N       0 MULT    MULTSEG              0   10   10 Y     N        Y      N
APOL14   IMS1           0          N       0 MULT    MULTSEG              0   10   10 Y     N        Y      Y
Trancode MbrName  Recover Resp Remote Serial WFI AOCMD Conv TranStat LclStat ModelName ModelType MSName
-------- -------- ------- ---- ------ ------ --- ----- ---- -------- ------- --------- --------- --------
APOL11   IMS1     Y       Y    N      N      N   N     N    N
APOL12   IMS1     Y       N    N      N      N   N     N    N
APOL13   IMS1     Y       N    N      N      N   N     N    N
APOL14   IMS1     N       N    N      N      N   N     N    N
Trancode MbrName  TimeAccess          TimeUpdate          TimeCreate          TimeImport          DefnType
-------- -------- ------------------- ------------------- ------------------- ------------------- --------
APOL11   IMS1                                             2008.267 12:32:05.43                    MODBLKS
APOL12   IMS1                                             2008.267 12:32:05.43                    MODBLKS
APOL13   IMS1                                             2008.267 12:32:05.43                    MODBLKS
APOL14   IMS1                                             2008.267 12:32:05.43                    MODBLKS
```

82

This is an example of the second page of the 'BYRSC' option printed output for the results of a type-2 QUERY TRAN NAME(*) SHOW(ALL) command.

This example shows printing all the information for the next 4 transactions in a way that is more readable than the WRAP method. Notice that the two columns, Trancode and MbrName, are repeated for each set of columns.

SPOC Print Options Support

- Benefits
  - Allows printing from SPOC to be more readable
- 3 print formatting options
  - WRAP
  - BYCOL
  - BYRSC
- Makes printing large command responses from the SPOC easier to use

83

The benefits of the new SPOC print options are that printed output can now be more readable and easier to use than previously. It improves the usability of SPOC printed output.

GSAM XRST Enhancement

- XRST pointing to incorrect (empty) GSAM output data set
  – Results in error message and user abend for job
- In previous IMS versions this error condition is not detected
  – Diagnosing condition can be difficult and result in data loss

The IMS 11 GSAM XRST support allows IMS to recognize the error scenario where a job is processing an XRST command during restart and incorrectly points to an empty output data set. In previous IMS versions, the IMS logic allows the program to point to an empty data set. After the restart IMS does not position the GSAM output to a point after the last committed record. IMS 11 returns an error message and the job abends allowing the user to correct the JCL and resubmit the job.

Problem Description

- GSAM XRST logic expects
  - Output data set is same data set prior to restart
- Problem occurs when output data set is Generation Data Group (GDG)
  - When GDG number is (+1) instead of (0)
    - Next GDG data set is used (empty data set)

In most cases before IMS 11, IMS expects an application doing a XRST restart for a GSAM database to use the same output data set that was used prior to the restart. However, given the way a Generation Data Group (GDG) data set is defined in a batch or BMP program, it is possible that the output data set during XRST restart will be the next GDG in line and will be empty. Generally, this is an undesirable situation during GSAM XRST processing.

Solution Description

- GSAM XRST logic will check for empty GSAM output data set
  - If detected
    - DFS1000I is issued
      - `DFS1000I IMSV11 ABEND U0102 MODULE=DFSZD210 LABEL=00001808 R2=C4C30001 R14=0003D8CA`
    - Job abends with U0102 (R2 = Reason Code = x'C4C30001')
  - Exception
    - If no output was created by original job
      - Job will be restarted successfully
      - If GDG(+1) used for restart
        - Empty data set exists in string of data sets

86

The GSAM XRST support allows IMS to check if the GSAM output data set is empty during XRST processing. If it is empty, then IMS will issue DFS1000I and will abend the job with a U0102 with reason code x'C4C30001'. The DFS1000I is an existing error message that is generated from module DFSZD210. It is being used for this error condition. The address given in R14 in the error message is the address in the dump that identifies the reason code of x'C4C30001'. This reason code shows that the error was due to an empty GSAM output data set. The 'LABEL=nnnnnnnn' is the offset into the IMS code where the error actually occurred.

There is one exception to this rule. If there was no output produced by the original job, then an empty data set is a valid outcome. When the job is restarted, it will complete successfully, and there will be an empty output data set in the string of GDG data sets.

Sample JCL

```
//JOBNAME   JOB
//****************************************************************
//*     PROGRAMS BMP USING CHKPT AND GSAM                         *
//****************************************************************
//BMP2 EXEC BMPUPROC,MBR=DFSDDLT0,PSB=LUSRSFB,IMSID=IMS1,CKPTID=LAST
//IMS      DD   DISP=SHR,DSN=USER.PSBLIB,UNIT=SYSDA
//         DD   DISP=SHR,DSN=USER.DBDLIB,UNIT=SYSDA
//GSAMTESI DD  DUMMY
//*SAMTESO DD UNIT=SYSDA,VOL=SER=USER05,DISP=(,CATLG,CATLG),
//*          SPACE=(CYL,(1,1)),DSN=GSAM.CONCATVB(+1)         <-Causes error
//GSAMTESO DD UNIT=SYSDA,VOL=SER=USER05,DISP=MOD,
//           SPACE=(CYL,(1,1)),DSN=GSAM.CONCATVB(0)          <-Correct GDG
//IMSLOGR  DD   DSN=USER.OLDSP0,DISP=SHR
//PRINTDD  DD   SYSOUT=A
//IMSERR   DD   SYSOUT=A
//SYSPRINT DD   SYSOUT=A
//DFSSNAP  DD   SYSOUT=A
//SYSIN    DD   *
S 1 1 1 1 11  1USRG1
L   0001 SNAP
WTOR  * DFSDDLT0 SAMPLE INPUT *
L        XRST
L      8 DATA                                               +
```

87

In this example, the GSAM output data set at label GSAMTESO is shown correctly and incorrectly. The correct way is to use the GDG(0) specification so that the GSAM output data set is reused during the XRST restart. The incorrect way is to use GDG(+1) since this allows IMS to create a new and empty GSAM output data set using the GDG naming convention of incrementing the name by ‘1’.

U0845 Diagnostics

- U0845 Abend is for Database System Errors
- New Message (DFS1058E) provides abend information
  - Quicker diagnosis of problem
  - Serviceability enhancement

The U0845 abend has been in IMS for many years. It is used to identify certain IMS system errors that can occur. There are five modules that can issue the abend and with an analysis of the registers at the time of the abend, IBM can determine the system error that occurred.

The U0845 enhancement allows a new message, DFS1058E, to accompany the U0845 abend. This message will have the same information that is in the abend.

U0845 Abend

- Description:
  - An unexpected error occurred in:
    - DFSDVSM0, DFSDVBH0, DFSTOCL0, DFSFXC50 or DFSNOTB0

Module	Description
DFSDVSM0	Interface between DL/I Action Modules and VSAM
DFSTOCL0	I/O Toleration DB Close
DFSDVBH0	DL/I Buffer Handler Router
DFSFXC50	Database Sync Point Processor
DFSNOTB0	DL/I Buffer Handler Notify Exit Processor

The U0845 abend can be issued by one of five modules: DFSDVSM0, DFSDVBH0, DFSTOCL0, DFSFXC50 and DFSNOTB0. These are DL/I modules that are at the heart of database processing. The U0845 condition is an internal database error. The new error message, DFS1058E, will make it easier and faster to diagnose these error conditions because the information contained in the U0845 abend will now be available in the DFS1058E error message on the system console.

Solution

- DFS1058E error message will accompany U0845 abend
  - Exception: DFSNOTB0
    - DFSNOTB0 runs in SRB mode
    - Messages can not be issued in SRB mode
    - For DFSNOTB0, (R6 = Error Reason Code, R1 = x’034D’ = d’0845’)

- DFS1058E - REASON = xxx – cccccccccccccccccccccccccccccccccc
  - xxx = reason code
  - ccc… = brief text message indicating the type of system error
  - Example:
    - DFS1058E – REASON = 010 - NON-VSAM BQEL - BUFFER NOT WRITTEN, CANNOT FREE BQEL

The solution to providing a faster diagnosis of the U0845 abend is to provide a new error message prior to the abend. The new message is DFS1058E and it contains the same information that could be derived from an analysis of the U0845 dump. The DFS1058E message contains a unique reason code and a short description of the error condition. The module, DFSNOTB0, runs in SRB Mode and this prevents IMS messages from being issued. For DFSNOTB0, register 1 will contain the U0845 abend number and register 6 will identify the reason code.

/DIAGNOSE Command Enhancements

- New options for the /DIAGNOSE command
  - /DIAGNOSE SNAP MODULE(modname)
    - Command response includes the entry point address and prolog information for the specified IMS module.
      - Entry point address can be used as input to the MVS SLIP command
      - Prolog information includes the maintenance level
  - /DIAGNOSE SNAP BLOCK(CSCD)
    - Allows the user to capture storage information for the APPC/OTMA Shared Message Queues SCD Extension control block
  - /DIAGNOSE SNAP STRUCTURE(structurename)
    - Command response includes storage information for the DFSSQS control block storage for the specified Shared Queues structure
- Benefits
  - Improved diagnostic information
  - Eliminates need for some dumps

91

The search process for the /DIA SNAP MODULE command accounts for (1) any module which resides in the IMS nucleus, (2) dynamically loaded modules, and (3) composite modules whose structure has been identified and internally mapped. The only modules in the third category are the /DIAGNOSE command processing modules.

The module entry point address is especially useful for SLIP invoked by MVS SLIP commands. The address extracted by the /DIAGNOSE command can be used immediately instead of having to first produce a dump.

The standard prolog information returned by the /DIA SNAP MODULE 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

/DIAGNOSE Command Syntax

```
                                          ┌─LENGTH(8)─────┐                       ┌─OPTION(OLDS)──┐
►►──┬─/DIAGNOSE─┬──SNAP───────┬──ADDRESS(address)─┼───────────────┼─┬─────────────┬──┬─┼───────────────┼──►◄
    └─/DIA──────┘             │                   └─LENGTH(length)┘  └─KEY(┬ 0 ┬)┘  │  └─OPTION(TRACE)─┘
                              │                                            └ 7 ┘    │
                              ├──BLOCK(┬ ALL  ┬)────────────────────────────────────┤
                              │        ├ CMDE ┤                                     │
                              │        ├ CSCD ┤                                     │
                              │        ├ ESCD ┤                                     │
                              │        ├ LSCD ┤                                     │
                              │        ├ MWA  ┤                                     │
                              │        ├ QSCD ┤                                     │
                              │        ├ SCD  ┤                                     │
                              │        ├ SQM  ┤                                     │
                              │        └ TSCD ┘                                     │
                              ├──LTERM(ltermname)───────────────────────────────────┤
                              ├──MODULE(modname)────────────────────────────────────┤
                              ├──NODE(nodename)─────────────────────────────────────┤
                              ├──STRUCTURE(structurename)───────────────────────────┤
                              ├──TRAN(tranname)─────────────────────────────────────┤
                              └──USER(username)─────────────────────────────────────┘
```


The new options have been added to the syntax of the /DIAGNOSE command.

Dump Formatter Enhancement

- Ability to write log records from a dump that are not on the log
  - IMS Dump Formatter can write records to new data set
    - Writes log records which were in buffers in memory
  - IMS Dump Formatter Panel Changes
    - New panel added to EDA formatter, SYS option
  - Process:
    - The dump formatter writes the final log records to a data set
      - Also writes a report
    - KBLA, DFSERA10, or other tools may be used to analyze log records
  - A batch job version is also available
- Benefit
  - Simplified analysis of log records
    - Records previously only available from dump are now easily accessible

93

Sometimes the final records on an IMS log are not physically written when a failure occurs. These records are in memory and are included in a dump of the address space. This enhancement allows users to write these final log records with a new dump formatter function. This function reads the final log records from the dump and writes them to a data set.

To use the enhancement you should choose SYS from the (Enhanced Dump Analysis) panel. This will take you to the Systems Formatting Options panel where you should select WRITE (write data to output data set) option. Then, you specify the data set names of the output data set and a report data set. Next, choose the LOGEX option from the Write Formatting Options panel.

If the specified data set name exists, then the dump formatter will open it and allocated it. If is does not exist, then the dump formatter will create it first before opening and allocating it. The following are data set guidelines:
The Output IMS Data Set should be a sequential data set with the following attributes:
LRECL(32756) BLKSIZE(32760) RECFM(VB) . .
The Output Report Data Set should be a sequential data set with the following attributes:
LRECL(133) BLKSIZE(1330) RECFM(FBA) .

The report data set contains log sequence numbers from the log records in the dump buffers. This corresponds to the information formatted in the LOG option from the High Level Formatting menu. Also included at the bottom of the report is the number of log records written to the data set. This can be cross checked with the number of log records formatted in the LOG option. A count of the log records formatted by the LOG option has been added to the formatter in conjunction with this enhancement.

The capability to write log records from the dump to a data set is also provided by a batch job in IMS Version 11.

This enhancement simplifies the analysis of log records by including all of them in the log data set. It eliminates the need to examine some records from the log data set and others from a dump.

The enhancement will be used primarily by IBM service but is available to all users.

Dynamic Abend Dump Exit

- IMS 11 automatically installs new dump format exit routine (DFSAFMX0)
  - Previous versions of IMS required manual installation of the IMS Dump Formatting routine (DFSAFMD0)
    - Bind DFSAFMD0 into SYS1.LPALIB or an MLPA library
    - Add DFSAFMD0 name to IEAVADFM CSECT of IGC08054A in SYS1.LPALIB
  - IMS 11 dynamically adds the new module (DFSAFMX0)
    - DFSAFMD0 is not used by IMS 11
- Migration consideration
  - Do not delete DFSAFMD0 from system while IMS 10 or previous versions of IMS are in use
- Benefit
  - Simplifies IMS installation process

94

IMS 11 eliminates the need to manually install the IMS dump formatting routine (DFSAFMD0). IMS 11 dynamically installs a new module, DFSAFMX0. This eliminates the requirement to add DFSAFMD0 to the z/OS system when installing a new IMS release.

With previous versions of IMS, a bind of DFSAFMD0 into SYS1.LPALIB or an MLPA library was required and DFSAFMD0 had to be added to the IEAVADFM CSECT of IGC0805A in SYS1.LPALIB. Each new version of IMS had a new version of the DFSAFMD0 module. These modules supported the release with which they were shipped and previous IMS releases. IMS 10 and previous releases still requires DFSAFMD0. IMS 11 does not require this module.

Migration consideration:

IMS 10 and earlier versions of IMS still require the DFSAFMD0 module. It should not be deleted from the system until IMS 10 or earlier versions are no longer used in the system.

LSQA Storage Reduction

- Reduced LSQA usage for storage management
  - Previous IMS releases created many CDE control blocks in LSQA
    - Used to track GETMAINed storage obtained by IMS
    - Storage is below the line
      - Can cause 40D abends (end-of-memory) forcing an IPL
  - IMS 11 uses IMS tracking structure in 64-bit storage
    - Used to track GETMAINed storage obtained by IMS for BCB control blocks
    - Implemented by change in IMS IMODULE GETMAIN and DELETE
      - Can be used only by the CTL and DLISAS regions
      - Does not affect user application coding

95

IMS provides an internal IMS service called IMODULE, which IMS modules use to allocate and release storage, and load and delete modules. In previous IMS versions 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). 24-bit private storage is a limited resource (limited to a maximum of sixteen megabytes, but more practically in the range of eight to ten megabytes). With the large size of today's address spaces, it is possible to allocate more storage areas in 31-bit storage than it is possible to track using 24-bit CDE structures. When this happens in the IMS CTL or DLISAS address spaces, it is often the case that z/OS itself cannot get enough storage to perform recovery/termination manager (RTM) processing for the address space. This leads to "end-of-memory" (EOM) type abends, where IMS is unable to cleanup its allocated common storage. This often requires a z/OS IPL to clear up the "orphaned" common storage so that IMS can be restarted on that z/OS.

This enhancement creates a new internal IMS service and control block structure for tracking storage. This does NOT affect user application coding. It only affects internal IMS code and any user or vendor code which takes advantage of the IMS facilities for acquiring storage. The use of this new tracking service for any given storage request is controlled by coding a new optional parameter on the IMODULE macro. The storage tracking elements (STEs) are built in 64-bit private storage for the CTL region and DLISAS region. Other region types continue to use the CDE tracking technique.

The IMODULE GETMAIN storage requests for an internal type of IMS block called a "BCB IPAGE" are changed to use the new parameter 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. Thus, moving this one type of IMS storage to be tracked by STEs should address many of the common scenarios that end up leading to EOM and z/OS IPL situations.

LSQA Storage Reduction

- Vendors and users have used IMODULE to acquire storage
  - They may scan the CDE control blocks currently created by IMODULE
    - The default in IMS 11 will use CDEs
    - Vendors and users may code TRACK=STE to reduce their usage of LSQA
      - TRACK=STE eliminates the possibility of scanning the CDEs to find storage
- Related storage enhancement
  - All 31-bit storage for IMS BCB and DFSPOOL blocks are defined as 31-bit storage only
    - They cannot wrap to 24-bit storage
- Benefits
  - Eliminates potential IPL outages for storage clean up

There is one complicating factor regarding changing IMS's use of z/OS CDEs to track IMODULE GETMAINed storage. While CDEs are not an intended interface, they are a "well known" interface. Vendor and customer code can scan the CDEs today to find a particular piece of storage. Thus, a wholesale removal of the tracking of GETMAINed areas from the CDE chains could break existing customer and vendor code that depends on the CDEs being there. As a result, a selective approach is taken by this enhancement regarding what storage will be removed from being tracked by CDEs. The default IMODULE behavior is to continue to track GETMAINed storage with CDEs. An IMODULE caller must code a new parameter – TRACK=STE – to get the new tracking. For this line item, only DFSBCB IPAGE storage will be changed to be tracked with STEs; however, future use of STE tracking – particularlly with new storage areas – will certainly occur.

All of the new tracking blocks and modules are OCO, per IBM's direction for new code. There is no support for direct scanning of the storage manager's internal blocks.

BCB and DFSPOOL are internal IMS control block storage. In previous releases, control blocks managed by these techniques could expand into 24-bit storage (below the line). IMS 11 changes this by limiting these control blocks to 31-bit storage (above the line). This limits the 24-bit storage that IMS will acquire, leaving more available for LSQA.

The benefits of these changes are obvious. By eliminating this use of LSQA which is limited to 24-bit storage some out-of-memory conditions will be eliminated. Since storage clean up may fail when out-of-memory conditions are encountered, they often force IPLs to clean up storage. This enhancement should eliminate the IPLs required by this use of LSQA.

EAV - Extended Address Volumes

- EAV support was added in z/OS V1R10
- Extended Address Volumes have more than 65,520 cylinders
  - Up to 262,668 cylinders; >55,689,379,200 bytes per volume
  - Any data set may reside on the first 65,519 cylinders
  - Only data sets with EAV support may reside on cylinder 65,520 or above
- VSAM has EAV support
  - IMS has VSAM support for EAV
    - Database data sets (KSDS and ESDS)
      - Fast Path and full function including HALDB
    - RECONs
  - IMS APARs for EAV support:
    - IMS Version 9: PK72529; IMS Version 10: PK72530; IMS Version 11: PK78388
- There is no support for OSAM data sets on cylinder 65,520 or above

97

z/OS V1R10 added support for Extended Address Volumes. These volumes may have more than 65,520 cylinders. In order to support data on cylinders above 65,519 z/OS changed the addressing scheme for these cylinders. Before this support an address on DASD was four bytes of the from CCCCHHHH where each character was a hexadecimal digit. CCCC addressed the cylinder. HHHH addressed the head or track within the cylinder. For cylinders past 65,519 z/OS changed the scheme for those cylinders to be CCCCcccH. ccc is the high-order part of the cylinder address. If ccc is all zeros, the address is interpreted as CCCCHHHH. This scheme works since modern storage systems emulate 3390s and, therefore, are limited to 15 tracks be cylinder. Only one hexadecimal digit is required for the track part of the address.

Data sets which reside past cylinder 65,519 require software to interpret the new addressing scheme. VSAM support was added in z/OS V1R10. IMS now supports VSAM data sets using these Extended Address Volumes. This applies to VSAM database data sets including Fast Path and full function. Full function includes HALDB. It also applies to thre RECONs.

The IMS support of EAV was done via the following APARs and PTFs:

IMS 9: APAR PK72529: PTF UK43019

IMS 10: APAR PK72530: PTF UK43020

IMS11: APAR PK78388; PTF UK45102

New User Exits

- Early Initialization
  - Called early in the IMS initialization process
  - Called after user exit refresh
    - Function code indicates that the exit is called after a user exit refresh
- CQS Event
  - Called when IMS receives notification of a CQS event, such as CQS termination
- CQS Structure Event
  - Called after IMS receives notification of a CQS structure event, such as a structure rebuild

98

Three new user exits are provided.

The Early Initialization exit is called early during the IMS initialization process. This is before the IMS restart exit. The Early Initialization exit is also called after the refresh of a user exit. Refresh is explained on a later page. The Early Initialization exit is available for DB/DC, DBCTL, DCCTL, and FDBR systems. IMS 11 and previous IMS versions also have an Restart Exit routine. The Restart Exit is available in DB/DC, DBCTL, and DCCTL environments.

The CQS Event exit is called when IMS receives notification of a CQS event. An example of an event is the termination of CQS.

The CQS Structure Event exit is called when IMS receives notification of a CQS structure event. An example of one of these is the rebuilding of a CQS structure.

These exit routines will probably be used primarily by IMS tools; however, they are available to all users.

User Exits Defined in DFSDFxxx

- Restart exit
  - Available in IMS 10
- Early Initialization, CQS Event, and CQS Structure Event
- Sample definition in DFSDFxxx
  - Some of the exit types have multiple exit routines:

```
<SECTION=USER_EXITS>
EXITDEF=(TYPE=EINIT,EXITS=(EINIT02,EINIT01,EINIT04))
EXITDEF=(TYPE=ICQSEVNT,EXITS=(CQEVNT01,CQEVNT02))
EXITDEF=(TYPE=ICQSSTEV,EXITS=(CQSEVN01,CQSEVN03))
EXITDEF=(TYPE=RESTART,EXITS=RST01)
```

99

The new exit routines are defined in the DFSDFxxx IMS PROCLIB member. The IMS Restart exit is also defined in this member.

These exits may be defined with multiple exit routines. This is shown in the example.

Capabilities for the Exits Defined in DFSDFxxx

- Multiple exit routines of each type may be defined
  – Called in the order of the specification in DFSDFxxx
- Exit routines may be refreshed
  – REFRESH command
- Information about these exit routines may be queried
  – QUERY command
- Benefits
  – Easier management and dynamic change capabilities for exit routines

100

When multiple routines are defined for an exit, they are called in the order in which they are specified in the DFSDFxxx member.

The exit routines defined in DFSDFxxx may be refreshed with the REFRESH command. Information about the exit routines may be obtained with the QUERY command.

Commands for User Exits

- QUERY USEREXIT command

```
QUERY USEREXIT TYPE(types) SHOW(values)
```

  – Types may be EINIT, CSQEVNT, CQSSTEV, and RESTART
  – Information returned by SHOW values:
    - ACTIVE - number of currently active instances of the exit routine
    - CALLS - number of calls to the exit routine since the last exit routine refresh
    - ENTRYPT - entry point address of the exit routine
    - LOADPT - load point address of the exit routine
    - ETIME - cumulative elapsed time spent in exit routine since the last refresh
    - RTIME - date and time the exit was last refreshed or loaded
    - SIZE - the size in bytes of the exit routine; shown in hexadecimal
    - TEXT - The 32 bytes starting from offset +04 from the exit module's entry point
      – Text is translated to EBCDIC with non-printable characters replaced by periods

The QUERY USEREXIT command returns information about these exit routines. The default routing is to all IMSs in the IMSplex.

The types that may be specified for the QUERY TYPE parameter are EINIT, CQSEVNT, CQSSTEV, and RESTART. An asterisk (*) may be specified to query all types. The values which are valid for SHOW are ACTIVE, CALLS, ENTRYPT, LOADPT, ETIME, RTIME, SIZE, TEXT, and ALL.

The TEXT value may be specified to show data from the exit routine. This address in the exit routine is a common location 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 returned.

Commands for User Exits

- REFRESH USEREXIT command

```
REFRESH USEREXIT TYPE(types)
```

  - Refreshes (reloads) the specified exit routines
    - Reads DFSDFxxx member to determine exit routine names
  - Deletes old exit routine modules
  - Types allowed:
    - EINIT, CQSEVNT, CQSSTEV , and RESTART

102

The REFRESH USEREXIT command refreshes the specified exit routines. The default routing is to all IMSs in the IMSplex.

The types that may be specified are EINIT, CQSEVNT, CQSSTEV, and RESTART.

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

1. Reads the DFSDFxxx member and process the USEREXITS section of the DFSDFxxx member.
2. Loads the user exit modules specified in the USEREXITS 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, the old modules will be deleted.

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.

When any exit type is refreshed, the RESTART exit is driven.

KBLA Enhancements

- Scrolling capability added to data entry panels
  - Split screens can hide lines on the panel
    - Scrolling provides access to these lines
- Output datasets may be defined as multiple volumes
  - 'Number of vols' parameter added to the 'Define KBLA Environment' panel
  - Avoids potential out of space (x37) abends
- Support added for new log records in IMS 11
- Benefits
  - Improved usability for KBLA

103

Knowledge Based Log Analysis has been enhanced in three ways.

1. A scrolling capability has been added to the data entry panels. When the lines from a panel are hidden due to splitting the screen, they may now be accessed by using this scrolling capability.

2. A new parameter has been added to the 'Define KBLA Environment Panel' to allow you to specify the number of volumes for output data sets. Previously, the generated JCL would use the default of one volume. For large output data sets, the specification of multiple volumes may avoid potential out of space abends.

3. The new log records added by IMS 11 may be processed by KBLA.

Examples of the first two enhancements are shown on the following pages.

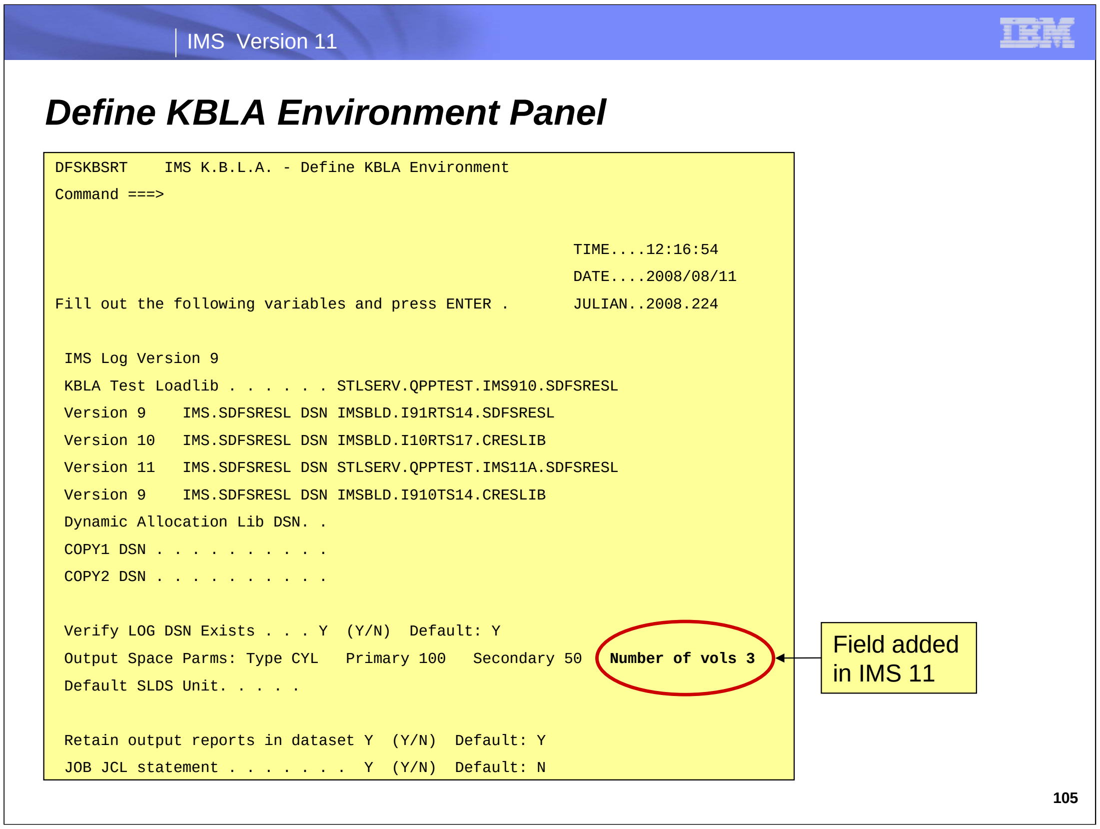
KBLA Data Entry Panel in Split-Screen

```
 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
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
```

Indicates that scrolling is available

104

This is an example of a KBLA panel with a split screen. All of the lines of the panel are not viewable on this screen. Scrolling may be used to view the other lines.



This is a Define KBLA Environment Panel. IMS 11 adds the 'Number of vols' parameter to the 'Output Spzce Parms:' line. The parameter defaults to 1. When another value is specified, output data set DD statements use the value for the fourth subparameter of the VOLUME= parameter. For example, the DD statement might be the following:

```
//SYSUT4     DD DSN=S840636.PM27429.KBLA.X08224.Y122710,
//              UNIT=SYSDA,
//              VOLUME=(,,,3),
//              SPACE=(CYL,(100,50),RLSE),
//              DCB=(RECFM=VB,LRECL=32756,BLKSIZE=32760),
//              DISP=(NEW,CATLG,CATLG)
```

System Enhancements

- DRD Export, Import, and Utilities
- ACBLIB Usability Enhancements
- SPOC Print Options (IMS 10 SPE)
- GSAM XRST Enhancement
- U0845 Diagnostics
- /DIAGNOSE Command Enhancements
- Dump Formatter Enhancement
- Dynamic Abend Dump Exit
- LSQA Storage Reduction
- Extended Address Volumes (EAV) Support
- New User Exits
- KBLA Enhancements

106

IMS Version 11

Security

Information Management software

RACF Enhancement:

Mixed-case Password Support

Impact to IMS

- DBC, DCC, and IMS procedures all contain a PSWDC= parameter to specify whether or not mixed-case password support is active
- In IMS 10 as originally delivered, available values are:
  - PSWDC=U (all passwords are forced to upper case)
    - Default value
  - PSWDC=M (mixed-case passwords allowed)
- IMS 11 and IMS 10 with an SPE add a third option
  - PSWDC=R
    - Uses RACF designation for mixed-case password support (RACF SETROPTS definition)
    - New default value
    - Will change when RACF changes (IMS automatically picks up new value)
      - No IMS restart required when changing to mixed-case password support

3

In IMS 11 and an SPE (Small Programming Enhancement) for IMS 10, an enhancement was made for RACF mixed-case password support. Each IMS system has a PSWDC= parameter which indicates whether mixed-case password support is active or not. It is important to note that in order to have mixed-case password support in IMS, RACF must first be set up with mixed-case password support.

In IMS 10 as it was originally delivered, the default value for this parameter was that all passwords would be converted to upper case (PSWDC=U). In IMS 11 and IMS 10 with the SPE, a new default value for this parameter has been added, PSWDC=R, which indicates that RACF will be used to determine whether mixed-case password support is active. If the value in RACF changes, IMS will automatically pick up this new value and does not require a restart.

The IMS 10 SPE is delivered in PTF UK46601 for APR PK80028.

Impact to IMS Connect

- HWSCFGxx is the IMS Connect configuration PROCLIB member that specifies environmental settings for IMS Connect
- PSWDMC= parameter specifies whether or not mixed-case password support is active
- In IMS 10 as originally delivered, available values are:
  - PSWDMC=N (all passwords are forced to upper case)
    - Default value
  - PSWDMC=Y (mixed-case passwords allowed)
- IMS 11 and IMS 10 with an SPE add a third option
  - PSWDMC=R
    - Uses RACF designation for mixed-case password support (RACF SETROPTS definition)
    - New default value
    - Will change when RACF changes

4

This change also applies to IMS Connect: the PSWDMC= parameter value defined in the HWSCFGxx member indicates whether mixed-case password support is active or not. Again, it is important to note that for IMS Connect to have mixed-case password support, RACF first must be set up to support mixed-case passwords (just like as is the case with IMS).

In IMS 10 as originally delivered, the default value was that all passwords were automatically converted to upper case. With IMS 11 and the IMS 10 SPE a new parameter value of PSWDMC=R was added, which means RACF will determine whether mixed-case password support is active by checking the RACF SETROPTS value. This is the new default value in IMS 11 and the IMS 10 SPE. When the R option is specified, if RACF changes from upper case to mixed case or back, IMS Connect will also make the change.

The IMS Connect SPE is PTF UK45982 for APAR PK80037.

Impact to IMS Connect

- IMS Connect CONFIG member (HWSCFGxx)
  - Add third option R to the PSWDMC= Y|N|**R** in HWS= statement

- IMS Connect commands
  - Add third option RCF

```
►►──SETPWMC──────ON──────────────────────►◄
                ├──OFF──┤
                 ──RCF──┘

►►─UPDATE──MEMBER──TYPE(IMSCON)──SET(PSWDMC(───ON────))──►◄
                                             ├─OFF─┤
                                              ─RCF─┘
```


In addition to PSWDMC=R being added to the IMS Connect CONFIG member, the IMS Connect commands that set this value have also been enhanced. The IMS Connect Type-1 command SETPWMC has a new possible value of “RCF” and the z/OS Modify Interface command UPDATE MEMBER for TYPE(IMSCON) has a new possible parameter value of “RCF” as well for the SET(PSWDMC)() value. The new syntax diagrams for both these commands are shown here with the new value in bold. In both of the commands, setting the value to “RCF” means that RACF will be checked to determine whether mixed-case password support is active or not.

Migration Considerations

- New default values in IMS 11 and IMS 10 SPE
  - PSWDC=R for IMS
  - PSWDMC=R for IMS Connect
    - Coordinate this setting between IMS and IMS Connect
- Recommendation: change value to default of ‘R’
  - PSWDC=M → PSWDC=R
  - PSWDMC=Y → PSWDMC=R

6

In IMS 11 and IMS 10 with the SPEs, it is important to note the new default values for the IMS and IMS Connect mixed-case support parameters. For both IMS and IMS Connect, the new default parameter checks RACF to determine whether mixed-case password support is in effect. It is also advised 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 10 without the SPE (with a setting of PSWDC=M for IMS or PSWDMC=Y for IMS Connect), the value should be changed to use RACF (=R) in IMS 11 or when the IMS 10 SPE is applied, since mixed-case parameter support needs to be active in RACF for activation anyway.

IMS Version 11

Benefits

- Mixed-case password support setting for IMS/IMS Connect now automatically consistent with RACF setting by default
  - When RACF changes to mixed-case password support:
    - No IMS restart required
    - No IMS Connect command required

7

The benefit of this capability is that IMS and IMS Connect will now automatically have the same parameter value for mixed-case password support by default. Without the enhancement, if RACF changes to support mixed-case passwords, each IMS system must be restarted with the new mixed-case parameter support designation on the PSWDC= parameter. With the new support of PSWDC=R, IMS does not need to be restarted in order to switch to using mixed-case password support when RACF changes.

In the case of IMS Connect, there is a command available to switch to using mixed-case passwords but if RACF changes to mixed-case passwords and the IMS Connect command isn't issued to switch to mixed-case, signon attempts will fail. In addition, restarting IMS Connect will result in the HWSCFGxx PROCLIB member being read and since there are no logs with IMS Connect, the mixed-case password support previously instated by the command will not persist across the restart.

So as you can see, the new default setting for mixed-case password support benefits the user by easing the process by which mixed-case password support is implemented.

SMU to RACF Conversion Aid Utilities

SPEs for IMS 9/10

Overview

- SMU security support removed after IMS 9
- After IMS 9, IMS security must be implemented with a security product that uses System Authorization Facility (SAF)
- Conversion Aid utilities created to facilitate migration from SMU security to RACF security
  - Delivered as IMS 9 SPE
  - Forward fit to IMS 10
  - Included in IMS 11 base

IMS 9 was the last release to support Security Maintenance Utility (SMU) security. Support was added in this release for any System Authorization Facility (SAF)-based security product. After IMS 9 was released, an SPE was made available containing Conversion Aid utilities that can be used to assist with converting from SMU security to RACF security. As you will see, these utilities provide a few different functions which result in reduced time and effort in the migration process. For example, they provide a roadmap and checklist that can be followed, as well as automation in converting large quantities of control statements. The Conversion Aid utilities were delivered as an IMS 9 SPE, which was forward fit to IMS 10 and included in the IMS 11 base code.

Conversion Aid Utilities Capabilities

- SMU control statements and Stage 1 macros are read to create:
  - Comparable RACF control statements
  - Changes to Stage 1 macros
  - Stage 1 Analysis Reports (checklists for conversion)
- No user exit changes are made
- Utilities run offline and can be run prior to migration
- Note: Not all changes required to implement RACF security are completed by utility -- **only the manual effort-intensive ones are**
  - For a complete checklist of steps, run a Stage 1 Analysis report

10

The primary purpose of the Conversion Aid is to help generate RACF resource profiles and user profiles from SMU data. In some cases the IMS Stage 1 may be optionally updated. For example, sign on for specific static terminals may update the TERMINAL macro and AOI Type 1 may update the TRANSACT macro. The utilities can also create checklists that can be referenced to ensure that all RACF migration steps are covered.

The utilities do not change any values in the STAGE1 SECURITY macro nor the DFSPBxxx, DFSDCxxx PROCLIB members, not does it generate any User Exit code. The utilities are run offline and can be run before migrating to RACF as a preparatory step. Please note that running these utilities will not cover all required steps to complete RACF migration – its purpose is to alleviate the steps that would require significant effort if done manually.

Conversion Aid Utility Prerequisites

- IMS 9 APARs/PTFs
  - PK68453/UK38824
  - PK66015/UK37339
  - PK56106/UK32791
  - PK54996/UK32790
  - PK38522/UK28607
  - PK35433/UK21894

- IMS 10 APARs/PTFs
  - PK69107/UK38825
  - PK66030/UK37313
  - PK56185/UK33359
  - PK58281/UK32794
  - PK49538/UK31516

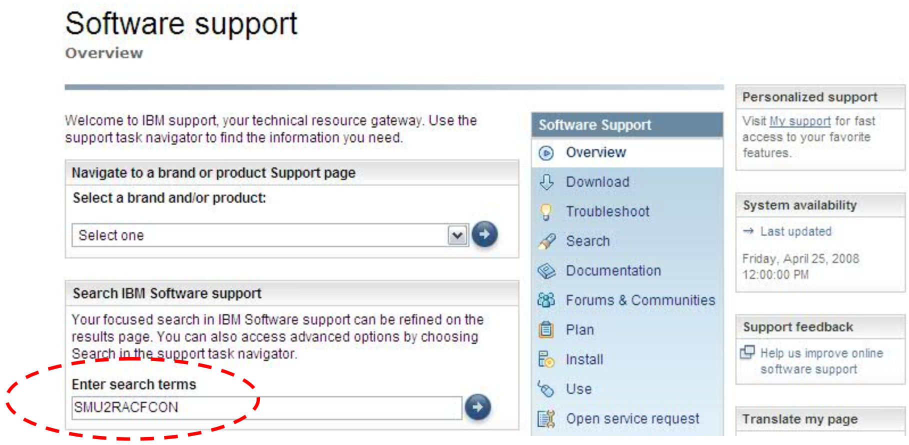
11

In order to use these utilities, these are the APARs that need to be applied. They are available for a IMS 9 or IMS 10 system.

IMS Version 11

Documentation

- Programs and JCL needed to run the utilities is documented in the *IMS System Utilities Reference* manual
- Documentation for the utilities is also in the PSP bucket (http://www.ibm.com/software/support):



12

To find the documentation for the SMU to RACF Conversion Aid Utilities, see the IMS System Utilities Reference manual, or go to http://www.ibm.com/software/support/

In the “Enter search terms” textbox, enter the search argument SMU2RACFCON. There are many run time options that enable the control the RACF output and it is advisable to read the documentation to get the most benefit from the Conversion Aid Utilities. The documentation also contains examples with sample input/output.

Benefits

- Time/effort required to convert from SMU to RACF reduced
  – Large volumes of control statements require conversion, which is automated by the utilities
- Utilities act as a ‘road map’ and checklist for conversion
- Assist with migration
  – IMS 8 to IMS 11
  – IMS 9 to IMS 11

13

The main benefit of the SMU to RACF Conversion Aid Utilities is the reduction in the amount of time and effort required in the RACF migration process. The utilities accomplish this by converting control statements that would otherwise need to be converted manually. The utilities also provide a roadmap and a complete checklist that can be followed.

The Conversion Aid utilities are useful in converting to IMS 11 from prior releases of IMS such as IMS 8 and IMS 9

SMU to RACF Conversion Utility

Reference and Examples

The following charts include a higher level of detail on the utilities, as well as example JCL and sample reports.

Stage 1 Analysis Reporting

- Stage 1 is read to generate a report
  - Provides guidance in performing SMU to RACF conversion. Parameters which are read from Stage 1, surmised from utility defaults and control statements are combined to prepare the report.
- The report consists of multiple sections
  - Resource Access Security
  - AOI Security
  - LTERM Security
  - TCO Security
  - LOCK, UNLOCK, SET
  - MSC Link Receive Security
  - Static Terminal Security
- Important: These reports can serve as conversion checklists, as they include all of the steps required for conversion, including some for which there is **no utility support** because the steps are not manual effort-intensive.

15

This utility reads Stage 1 to generate a report to provide some guidance in performing the SMU to RACF conversion. Parameters which are read from Stage 1, surmised from utility defaults and control statements are combined to prepare the report.

SMU to RACF Syntax conversions

- Resource Access Security
  -)(AGN (with AGLTERM, AGPSB, AGTRAN options)

    RACF TIMS, GIMS, IIMS, JIMS, LIMS, MIMS Resource Class conversion
- Logical Terminal Security
  -)(TERMINAL (COMMAND and TRANSACT options)

    RACF DIMS, CIMS, TIMS Resource Class conversion
  -)(COMMAND (TERMINAL options)

    RACF LIMS, CIMS, Resource Class conversion
  -)(TRANSACT (TERMINAL options)

    RACF LIMS, TIMS, Resource Class conversion
- AOI Security
  -)(CTRANS and)(TCOMMAND

    RACF DIMS, CIMS Resource Class conversion

16

This slide shows the different SMU statements and the RACF classes that they are converted to once the appropriate utility is run. The utilities impact 3 types of security: resource access, logical terminal and AOI security. We now explore these different areas of security and how the utilities relate to them.

Resource Access Security:
)(AGN Conversion Utility Sample JCL

```
//RUNAGN0  EXEC  PGM=DFSKAGN0
//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.DFSKAGN0.RACFFILE
//USERS    DD *
USER1 GROUP2
USER2 GROUP2
USER3 GROUP2
USER4 GROUP1
USER5 GROUP3
//CONTROL DD *
DELUSER
ADDUSER
CONNECT
//SYSPRINT DD SYSOUT=*
```

17

Here is some example JCL that will convert)(AGN SMU statements to RACF user groups. The program used is DFSKAGN0.

Resource Access Security:
)(AGN Conversion Utility Sample Report

```
DFSKAGN0 CONTROL: DELUSER
DFSKAGN0 CONTROL: ADDUSER
DFSKAGN0 CONTROL: CONNECT

1  DFSKAGN0: AGN SMU/RACF CONVERTER REPORT
                    DATE: 2007/086  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
```


Here is an example of the report generated after the)(AGN Conversion Utility is run. Note the data returned in the report output.

Resource Access Security:
)(AGN Conversion Utility Sample SMU Before Conversion

```
)(   AGN   IMSDGRP
     AGPSB    DEBS
     AGPSB    APOL1
     AGTRAN   DEBSTRN1
     AGTRAN   APOL12
     AGLTERM IMSUS02
     AGLTERM T3270LD

ADDUSER BMPUSER1
RDEFINE AIMS IMSDGRP OWNER(IMSADMIN) UACC(NONE)
PERMIT IMSDGRP CLASS(AIMS) ID(BMPUSER1) ACCESS(READ)
SETROPTS CLASSACT(AIMS)
```

19

This example illustrates SMU AGN control statements for a BMP region that will be converted to comparable RACF statements when the)(AGN Conversion Utility is run – see this chart for the SMU AGN statements and the next chart for the RACF equivalents. This example uses a BMP region since it can protect all three kinds of resource types. Before going to the next slide, take note of the PSB, TRAN and LTERM resource names that will be included in the utility processing. Also note the BMPUSER1 ID being permitted with read access for the resources defined in the IMSDGRP group defined above. Now for the RACF conversion…

Resource Access Security:
)(AGN Conversion Utility Sample RACF After Conversion

```
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)
```

- Green: Values taking default, or input as control statements
- Blue: Value may be supplied from USERS DD input
- Red: PERMIT statements which indicate which ID is authorized to use which component

After the SMU statements on the previous slide are run through the)(AGN Conversion Utility, here is an example of what the generate RACF statements would look like. Notice all of the PSBs, TRANs, and LTERMs included in the previous SMU statements are now defined to RACF with their respective RACF IMS classes. Also note that there are different users (denoted in green) that are being granted read access to the resources defined in the IMSDGRP group. Lastly, note the BMPUSER1 ID being permitted to access the resources defined in IMSDGRP, denoted in blue. Lastly, the PERMIT statements are coded in red since they are the most useful in detecting the users that are authorized to access the resources in specific classes.

AOI Security:
)(CTRANS and)(TCOMMAND Option Conversion Utility Sample JCL

```
//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.DFSKAGN0.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)
```

21

AOI Security:)(CTRANS and)(TCOMMAND Option Conversion Utility Sample Report

```
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
```

AOI Security:)(CTRANS and)(TCOMMAND Option Conversion Utility Sample Conversion

SMU

```
)( CTRANS AUTOCTL
     TCOMMAND START
     TCOMMAND STOP
```

RACF

AOI=TRAN produces:

```
ADDUSER  AUTOCTL  OWNER(*OWNER* ) DFLTGRP(*DEFGRP*)
PERMIT STA      CLASS(CIMS    ) ID(AUTOCTL ) ACCESS(READ  )
PERMIT STO      CLASS(CIMS    ) ID(AUTOCTL ) ACCESS(READ  )
```

AOI=CMD produces:

```
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  )
```


These examples illustrate SMU control statements and the comparable RACF statements generated by the conversion utility.

AOI Security:
)(CTRANS and)(TCOMMAND Option Conversion Utility Sample Conversion with AOI=

Pre-Conversion

```
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
```

Post-Conversion

```
* AOI= PARAMETER ADDED TO TRANSACT  SMU/STAGE1 CONV.
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
```

24

Logical Terminal Security:)(TERMINAL and)(TCOMMAND Option Conversion Utility Sample JCL

```
//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)
```

Logical Terminal Security:)(TERMINAL and)(COMMAND Option Conversion Utility Sample Report

```
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
```

Logical Terminal Security:)(TERMINAL and)(COMMAND Option Conversion Utility Sample Conversion

SMU Control Statements

)(TERMINAL TERM1
COMMAND START
COMMAND STOP
TRANSACT STATTRN

RACF Control Statements

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)

27

The following examples illustrate SMU control statements and the comparable RACF statements generated by the conversion utility where LTERMS are defined as USERs.

IMS Stage 1 SIGN Conversion: Examples of Stage 1

Pre-Conversion

```
          TERMINAL  NAME=TERMINL1,MODEL=2
              NAME  TERM1
          TERMINAL  NAME=TERMINL2,MODEL=2,                       X
               OPTIONS=(FORCRESP,,PAGDEL),                       X
               EDIT=(,YES),FEAT=(PFK,CARD,PEN),MSGDEL=NONIOPCB
              NAME  TERM2
```

Post –Conversion

```
* SIGNON PARAMETER ADDED TO TERMINAL OPTIONS  SMU/STAGE1 CONV.
          TERMINAL  NAME=TERMINL1,MODEL=2,OPTIONS=SIGNON
              NAME  TERM1
* SIGNON PARAMETER ADDED TO TERMINAL OPTIONS  SMU/STAGE1 CONV.
          TERMINAL  NAME=TERMINL2,MODEL=2,                        X
               OPTIONS=(FORCRESP,,PAGDEL,SIGNON),                 X
               EDIT=(,YES),FEAT=(PFK,CARD,PEN),MSGDEL=NONIOPCB
              NAME  TERM2
```

28

IMS Version 11

Database

Information Management software

Database Enhancements

- Database Quiesce
- HALDB Online Reorganization enhancements
  - Performance improvements
  - Statistics and restart enhancements
- IRLM Locktime enhancement
- Fast Path
  - 64 Bit Buffer Manager
  - OPEN option for UPD DB and UPD AREA commands
  - Unique Subcode for PROCOPT=GO U1026 Abends
  - Non-Recoverable DEDBs with SDEPs (SPE for IMS 10)
  - Access Type by Area (SPE for IMS 9 and IMS 10)

Database Quiesce

Database Quiesce

- Command to quiesce databases, areas and/or partitions
  - Creates a recovery point
    - A point to which a database may be recovered
    - OLDS is switched
    - DBRC ALLOC records are closed
    - Coordinated across IMSplex
    - All databases, areas and partitions have the same recovery point time
  - Databases are not taken offline
  - Transactions and BMPs are not terminated
    - Update activity is internally quiesced after the next application synch point

```
UPDATE DB|AREA|DATAGRP  NAME(*|name1,name2,…,namen)
       START(QUIESCE) OPTION(HOLD|NOHOLD  FEOV|NOFEOV)
       SET(TIMEOUT(nnn))
```

4

The DB Quiesce function is a new option on the type-2 UPDATE command. It allows a coordinated Recovery Point (RP) to be created across the IMSplex for one or more database data sets. A recovery point is a time when there are no DBRC ALLOC records with start times before the recovery point and end times after it. This allows the database to be recovered to this recovery point time. The OLDS is switched at the quiesce time. Recovery points are needed for recovery for some errors. If an operational or programming error causes incorrect updates to a database, it may be restored to the time of the quiesce by doing a timestamp recovery to this time. DBRC enforces the restriction that timestamp recoveries may only be done to times which are recovery points.

The UPDATE command with the START(QUIESCE) specification is used to quiesce one or more databases, HALDB partitions, DEDB areas, or data groups. Data groups may be database groups or database data set groups.

Databases are not taken offline by database quiesce. The database data sets are not closed or deallocated.

Transactions and BMPs which access the databases are not terminated for the quiesce function. When DB Quiesce is invoked, the command waits for all updates to the specified databases, partitions, or areas that are currently in progress to be committed and hardened to DASD. This is a point in time when the databases are consistent and there are no uncommitted updates. The DB Quiesce will wait for any applications which have not yet committed their updates to these databases and these applications will be allowed to access the databases until their commits are completed. Once the application has reached the commit point, the next DL/I update or get hold call results in a wait until the quiesce command has completed and has released the database. All new applications that are scheduled after the quiesce has started will be held up on their first DL/I update or get hold call to a database include in the command.

Database Quiesce

- DB Quiesce with HOLD
  - Two variations of Quiesce
    - Quiesce and NOHOLD
      - Database is quiesced and then immediately released from quiesce
    - Quiesce and HOLD
      - Database activity is quiesced until specifically released
      - Allows a clean image copy to be taken
- Command is less disruptive than /DBR and /DBD processing
  - Transactions and BMPs are held up on first update or get hold call after commit
    - Transactions and BMPs are resumed:
      - After QUIESCE and NOHOLD
      - After STOP(QUIESCE) for previous QUIESCE and HOLD

5

The quiesce created by the DB Quiesce command can be released immediately after the RP is created or it can be held until specifically released by another UPD command. When the RP is completed, the DBRC RECON data set ALLOC record is updated with a deallocation timestamp and a QUIESCE flag is written for each database listed in the DB Quiesce command.

The UPD … START(QUIESCE) command with the HOLD option is useful for creating a clean image copy because all update database activity is quiesced until an UPD command to stop the quiesce is issued. The UPD … START(QUIESCE) command with the NOHOLD option is useful for creating a RP without causing the database unavailability that the HOLD option creates.

The UPD … START(QUIESCE) command allows IMS to create a RP without having to issue /DBR or /DBD commands that deallocate database data sets. In this way, it is a less disruptive way of creating a RP.

Scope and Prerequisites

- DB Quiesce Scope
  - Database is quiesced across IMSplex
    - No local function (recovery points are always created for the IMSplex)
- CSL Use
  - SCI and OM are required
  - RM may be used
    - RM is required with multiple IMS systems
      - RM coordinates DB Quiesce process in IMSplex
    - RM is not required with a single IMS system
      - RMENV=N is valid
    - There is no global database status stored in RM for DB Quiesce
- RECON MINVERS value must be '11.1'
  - CHANGE.RECON MINVERS('11.1')

6

When one or more databases are quiesced in an IMSplex, the databases are quiesced for all IMS subsystems in the IMSplex. There is no local DB Quiesce function because there is no concept of a local Recovery Point in an IMSplex.

The command used for database quiesce is a type-2 command; therefore, SCI and OM are required. The Resource Manager (RM) is not required when there is only one IMS system. RM is required when there are multiple IMS systems registered to SCI. RM is used for DB Quiesce to coordinate a common quiesce recovery point across multiple IMS systems in the IMSplex. If there is only one IMS system, then RMENV=N is a valid option and RM is not used.

An RM structure is not required for DB Quiesce as there is no DB Quiesce status stored in the RM structure. However, since the RM structure coordinates the DB Quiesce process across the IMSplex, then if the RM address space goes down, the DB Quiesce process can be restored from the RM structure when the RM address space is available again. If there are multiple IMS systems in the IMSplex, then if RM is unavailable, the DB Quiesce command will be rejected with a RC=10, Reason Code = 4208.

If there is an IMSplex, the command always applies to all IMS systems in the IMSplex. A command master IMS system coordinates the DB Quiesce command in an IMSplex. If the DB Quiesce command is routed to more than one IMS subsystem in the IMSplex, then one IMS system is chosen to perform as the command master. The command master updates the DBRC RECON data sets for all participants in the quiesce function.

The DB Quiesce function does not have a Global database status that is stored in RM. If a Global database status has been defined in RM, then the DB Quiesce function does not affect this status in RM. The quiesce status of a database or area is returned to the IMS subsystem during database authorization. An update request is held up until the DB Quiesce is completed. A read request is allowed.

The DB Quiesce command function requires that the minimum version parameter, MINVERS, in the DBRC RECON data set be set to 11.1. The DBRC CHANGE.RECON MINVERS command will set this parameter value in the DBRC RECON data set. The DB Quiesce function requires the same DBRC RECON data sets across the IMSplex. The RP will only be consistent within the same set of DBRC RECON data sets.

Command Syntax - UPD … START(QUIESCE)

- Starting DB Quiesce:

```
UPDATE DB|AREA|DATAGRP NAME(*|name1,name2,…,namen)
       START(QUIESCE) OPTION(HOLD|NOHOLD  FEOV|NOFEOV  ALLRSP)
       SET(TIMEOUT(nnn))
```

- Default for NOHOLD is FEOV; Default for HOLD is NOFEOV
- SET(TIMEOUT(nnn))
  - nnn = 1-999 seconds
  - Time to wait for applications to commit before aborting the command
    - If any database cannot be quiesced, none are quiesced
  - Default may be set with DBQUIESCETO in DFSCGxxx or DFSDFxxx
    - Default is 30 seconds

```
<SECTION=COMMON_SERVICE_LAYER>
CMDSEC=N            /* NO CMD AUTHORIZATION CHECKING */
IMSPLEX=PLEX1       /* IMSPLEX NAME                  */
DBQUIESCETO=45      /* DB QUIESCE TIMEOUT            */
```

7

The DB Quiesce function is invoked by using the Type-2 UPD command. The UPD DB command is used for HALDB Master databases, HALDB Partitions and for DEDBs. The UPD AREA is for DEDB Areas. The UPD DATAGRP command is for a database or data set group.

There are two forms of the UPD … START(QUIESCE) command and they differ based on the OPTION(HOLD or NOHOLD) specification. The default is OPTION(NOHOLD) which implies that the quiesced databases are immediately released once the command is completed and the quiesce timestamp is recorded in the DBRC RECON data set. The second form of the command uses OPTION(HOLD). With this command, the databases are held in the quiesced state until they are specifically released by a subsequent UPDATE … STOP(QUIESCE) command. While the databases are held in the quiesced state, the image copy utility can run to create clean image copies.

The OPTION(FEOV) keyword is not a new option but it does apply to the Quiesce command. When OPTION(FEOV) is specified, the OLDS are switched at the completion of the command. By default, the OLDS are switched when OPTION(NOHOLD) is specified. When OPTION(HOLD) is specified, the OLDS are not switched by default until the UPD … STOP(QUIESCE) command is issued to release the quiesced databases. The OPTION(FEOV) or OPTION(NOFEOV) options are used to alter the default switching of the OLDS. ALLRSP may be specified with NAME(*).

The “SET(TIMEOUT(nnn))” keyword allows the installation to set a maximum time value for the command to complete. The command will fail if it has not completed for the specified databases by the time this value is reached. The ‘nnn’ time value is specified in seconds and can be 1-999. This function only applies to the START(QUIESCE) function. The time value specified applies to all of the databases in the command, not to each one individually.

If the timeout value is not specified in the command, it defaults to the value specified in the DBQUIESCETO keyword in the DFSCGxxx or DFSDFxxx PROCLIB member. If the DBQUIESCETO value is not specified, it defaults to 30 seconds.

DBRC Flags Effects on Command

- Database Status
  - If database has 'recovery needed,' backout needed' or 'reorg intent' flag set
    - UPD … START(QUIESCE) command cannot be initiated

- HALDB Online Reorganization
  - If OLR is active for a partition
    - Cursor is active
    - Owner exists for OLR for a partition
    - UPD … START(QUIESCE) command cannot be initiated
  - OLR cannot be started
    - If HALDB partition is 'QUIESCE HELD'

An UPD … START(QUIESCE) command cannot be issued against a database or area that is in a state where recovery or backout are needed, or if the reorganization intent flag is set. Similarly, an UPD … START(QUIESCE) command can only be issued for a HALDB partition that is not involved in HALDB Online Reorganization (OLR). If an UPD … START(QUIESCE) command is in progress or in effect for a HALDB partition, OLR can not be initiated for the partition.

Command Example - UPD … START(QUIESCE)

- UPD … START(QUIESCE)

```
   File  Action  Manage resources  SPOC  View  Options  Help

 PLEX1                        IMS Single Point of Control
 Command ===>

                            -   Plex . . PLEX1  Route . .
Wait . . 5:00
 Response for: UPD DB NAME(DEDBJN21) START(QUIESCE) OPTION(HOLD)
DBName   MbrName    CC
DEDBJN21 IMS1        0
DEDBJN21 IMS2        0
```

9

This is an example of an UPDATE DB … START(QUIESCE) command that starts a quiesce for a Fast Path DEDB. In this case, there are two IMS systems in the IMSplex and the quiesce is completed on both IMS subsystems with a return code of 0. The OPTION(HOLD) was specified so that the quiesce option would remain in affect until it was released by a subsequent UPDATE DB … STOP(QUIESCE) command.

Command Syntax - UPD … STOP(QUIESCE)

- Releasing a Held Quiesced Database:

```
UPDATE DB|AREA|DATAGRP  NAME(*|name1,name2,…,namen)
       STOP(QUIESCE)
       OPTION(FEOV|NOFEOV)
```

  – OLDS are switched unless OPTION(NOFEOV) is specified

The command to release a database that was held by an UPDATE … START(QUIESCE) OPTION(HOLD) command is the UPDATE … STOP(QUIESCE) command. By default, the OLDS are switched following the STOP(QUIESCE) command, but this can be altered if the OPTION(NOFEOV) option is selected.

Command Example - UPD … STOP(QUIESCE)

- UPD … STOP(QUIESCE)

```
   File  Action  Manage resources  SPOC  View  Options  Help

 PLEX1                    IMS Single Point of Control
 Command ===>

                        -   Plex . . PLEX1  Route . .
Wait . . 5:00
 Response for: UPD DB NAME(DEDBJN21) STOP(QUIESCE)
DBName   MbrName    CC
DEDBJN21 IMS1        0
DEDBJN21 IMS2        0
```


This is an example of an UPDATE DB … STOP(QUIESCE) command that releases a previously held DEDB on two IMS subsystems in an IMSplex. The command completes with Return Code 0.

Command Syntax - QUERY and /DIS

- QUERY for quiesced or quiescing status
  - 'QUIESCED' status exists only with HOLD
  - 'QUIESCING' status my exist with either HOLD or NOHOLD

```
QUERY DB|AREA|DATAGRP NAME(name)
      SHOW(ALL | STATUS)
      STATUS(QUIESCED | QUIESCING)
```

- /DISPLAY for quiesced or quiescing status

```
/DIS AREA areaname | DB dbname QSC
```


The existing Type-2 QUERY command is used to display the quiesce status of a database. The statuses associated with database quiesce are 'quiesced' and 'quiescing'. "Quiesced' means the database, area, or data group is currently queisced with HOLD. 'Quiescing' means the database, area, or data group is in the process of being quiesced. These statuses are shown in the response to the QUERY command when they exist and either ALL or STATUS is specified in the SHOW parameter. The two options STATUS(QUIESCED) and STATUS(QUIESCING) can be used as a filter to show the databases, areas or data groups which currently have the 'quiesced' or 'quiescing' status.

The QUIESCED status shows databases that have already been quiesced with the OPTION(HOLD) keyword. The QUIESCING status shows one or more databases that are currently being quiesced.

The type-1 /DISPLAY AREA and DB command may also be used to display the quiescing or quiesced status. The QSC keyword is a filter for these commands to show which databases or areas are quiesced or in the process of being quiesced.

Command Example - QRY … SHOW(STATUS)

- QRY … SHOW(STATUS)

```
   File  Action  Manage resources  SPOC  View  Options  Help

  PLEX1                    IMS Single Point of Control
  Command ===>

                    -   Plex . . PLEX1  Route . .   Wait . . 5:00
  Response for: QRY DB NAME(DEDBJN21) SHOW(STATUS)      More:   +
 DBName   AreaName MbrName    CC TYPE     LclStat
 DEDBJN21          IMS1        0 DEDB     QUIESCED
 DEDBJN21 DB21AR0  IMS1        0 AREA     QUIESCED
 DEDBJN21 DB21AR1  IMS1        0 AREA     QUIESCED
 DEDBJN21          IMS2        0 DEDB     QUIESCED
 DEDBJN21 DB21AR0  IMS2        0 AREA     QUIESCED
 DEDBJN21 DB21AR1  IMS2        0 AREA     QUIESCED
```

13

This is an example of a QUERY DB … SHOW(STATUS) command that shows a DEDB with all of its areas quiesced.

Command Example - /DIS … QSC

- /DIS DB QSC

IMS1	DATABASE	TYPE	... ACC	CONDITIONS
IMS1	AUTODB	DL/I	UP	NOTOPEN, QUIESCING
IMS1	DBHIDJ03	PHIDAM	UP	
IMS1	POHIDJC	PART	UP	NOTOPEN, QUIESCED
IMS1	DBOHIDK5	PHIDAM	UP	QUIESCED
IMS1	POHIDKA	PART	UP	NOTOPEN, QUIESCED
IMS1	POHIDKB	PART	UP	NOTOPEN, QUIESCED
IMS1	POHIDKC	PART	UP	NOTOPEN, QUIESCED

- /DIS DB AUTODB

IMS1	DATABASE	TYPE	... ACC	CONDITIONS
IMS1	AUTODB	DL/I	UP	NOTOPEN, QUIESCING

14

The first example shown here is of the /DIS DB command with the QSC filter. The response shows that the non-HALDB database AUTODB is quiescing, partition PHOIDJC of HALDB database DBHIDJ03 is quiesced, and all partitions of HALDB database DBOHIDK5 are quiesced.

The second example shows the response for a /DIS DB dbname command when the AUTODB database is quiescing. The QUIESCING status is in the CONDITIONS column.

Both examples are using the OM interface; therefore, the IMS ID, IMS1, is shown on the left of each line.

Command Syntax and Example - /DIS ACTIVE

- /DISPLAY ACTIVE
  - Example after quiesce command:

    UPD DB NAME(DHVNTZ02,DIVNTZ02) START(QUIESCE) OPTION(HOLD)

```
REGID  JOBNAME     TYPE  TRAN/STEP PROGRAM  STATUS   CLASS    IMS1
       2 MPPI3     TP    SHF1      PMVAPZ12 WAIT-QSC 1,   2   IMS1
       1 MPPI2     TP    NQF1      PMVAPZ12 WAIT-QSC 1,   2   IMS1
         JMPRGN    JMP   NONE                                 IMS1
       3 BMPI4     BMP             PMVAPZ12 WAIT-QSC          IMS1
         JBPRGN    JBP   NONE                                 IMS1
         FPRGN     FP    NONE                                 IMS1
         DBTRGN    DBT   NONE                                 IMS1
         DBRICSAK  DBRC                                       IMS1
         DLIICSAK  DLS                                        IMS1
   *08030/132621*                                             IMS1
```

15

The /DISPLAY ACTIVE command will show if an IMS region or thread is waiting due to a quiesce being held. This is shown with WAIT-QSC in the STATUS column.

This is an example of the /DIS ACTIVE REGION command. The REGION keyword on the /DIS ACTIVE command limits the response to the information about regions. The DB Quiesce command with the HOLD option has been issued. The BMP (BMPI4) and the two MPP regions (MPPI2, MPPI3) are attempting to do updates to the quiesced databases. Since the databases are being held, the regions show the status WAIT-QSC. The updates will complete when the UPD command with STOP(QUIESCE) is issued.

Batch Jobs Effects on Commands

- Databases cannot be quiesced when batch update jobs are running
  – Batch application with update intent causes UPD … START(QUIESCE) command to fail
- Databases can be quiesced when batch read jobs are running
  – Batch application with only read intent allows UPD … START(QUIESCE) command to succeed

16

If a batch application is running when an UPD … START(QUIESCE) command is issued, the result of the command will depend on the authorization of the batch application. If the batch application has update intent for the databases or areas in the command, the command will fail. If it has only read intent, the quiesce command will succeed.

DBRC RECON Records

- Two new flags in RECON database record
  - QUIESCE IN PROGRESS
    - New authorizations with read intent for database are allowed
    - New authorizations with update intent for database are not allowed
    - All utilities requesting authorization to database will fail
  - QUIESCE HELD
    - New authorizations with read intent for database are allowed
    - New authorizations with update intent for database are not allowed
    - All utilities (except Image Copy) requesting authorization to database will fail
- ALLOC records
  - Quiesce closes the ALLOC records for affected database data sets
    - Timestamp is the recovery point time
    - Flag added to ALLOC to indicate it was closed by quiesce

17

There are two new flags in the DBRC RECON database record. They are: 1) QUIESCE IN PROGRESS and 2) QUIESCE HELD. When the QUIESCE IN PROGRESS flag is set, new read authorizations are allowed for the database from an online IMS subsystem or batch job but new update authorizations are not allowed. All utilities will fail authorization including the image copy utility.

When the QUIESCE HELD flag is set, new authorizations are allowed for the database from an online IMS subsystem or batch job provided the request is for reading and not updating. All utilities will fail authorization with the exception of the image copy utility.

The Recovery Point timestamp is recorded as the deallocation timestamp for an ALLOC record in the DBRC RECON data set.

For Fast Path, a new ALLOC record is created in the DBRC RECON data set for all open DEDBs and areas. An ALLOC is not created for full function databases when the quiesced database is released. Instead, a new ALLOC record is created when the first update access is performed for the database. When the database quiesce is released, the dependent regions which were waiting at the first DL/I call will be started and can begin accessing the database.

DBRC LIST.DB and LIST.DBDS Commands

- Output of LIST.DB command has added information
  - 'QUIESCE IN PROGRESS'
  - 'QUIESCE HELD'
- Output of LIST.DBDS command for areas has added information
  - 'QUIESCE IN PROGRESS'
  - 'QUIESCE HELD'
- Output for ALLOC record has added information
  - "QUIESCE'
    - If present, the deallocation timestamp was added by a quiesce command

```
ALLOC
  ALLOC   =08.169 16:03:41.411778      *  ALLOC LRID =0000000000000000
  DSSN=0000000001 USID=0000000002 START = 08.169 15:59:33.231012
  DEALLOC =08.169 16:03:48.747189        DEALLOC LRID =0000000000000000
  QUIESCE
```

18

The DBRC LIST.DB and LIST.DBDS command output includes quiesce information. 'QUESCE IN PROGRESS' and 'QUIESCE HELD' states are listed as either 'YES' or 'NO' in the output of these commands. When 'QUIESCE HELD =YES' then 'QUIESCE IN PROGRESS =YES' is always present.

The output of LIST.DBDS commands includes ALLOC records. When the ALLOC was closed by a quiesce, 'QUIESCE' is included in the listing for the database data set. An example of this is shown on this page.

DBRC LIST.DB Example

```
DB
  DBD=DBHDOJ01                     IRLMID=*NULL        DMB#=1
TYPE=IMS
  SHARE LEVEL=3                   GSGNAME=**NULL**     USID=0000000001
  AUTHORIZED USID=0000000000  RECEIVE USID=0000000000 HARD
USID=0000000000
  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   =2
    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
```

19

This is an example of the DBRC LIST.DB command. It shows that the database for this DB record has been quiesced with the “HOLD” option. It should be noted that when the QUIESCE HELD flag is set to YES, the QUIESCE IN PROGRESS flag is also set to YES.

GENJCL.IC

- JCL for Image Copy must specify DISP=SHR when database is in 'QUIESCE HELD' state
  – Database is allocated to online system(s)

- GENJCL.IC
  – Generates DISP=SHR when database is 'QUIESCE HELD'
  – Generates DISP=OLD when database is not "QUIESCE HELD'
  – Keywords may be used to override DISP=
    – DBQUIH generates DISP=SHR
    – NODBQUIH generates DISP=OLD
    - Useful when GENJCL.IC is issued before UPD command but JCL will be used after the UPD command

20

When an UPDATE … START(QUIESCE) OPTION(HOLD) is issued against a database or area, the quiesce status is not released until an UPDATE … STOP(QUIESCE) command is issued against the same database or area. By holding the database or area in the quiesced state, a clean image copy can be run. Since the database data sets remain allocated to the IMS system, DISP=SHR must be specified for the database data set on the image copy job. GENJCL.IC has been updated to account for this.

By default the GENJCL.IC command will create a DD statement with DISP=SHR when the QUIESCE HELD flag exists for the database data set at the time the JCL is generated. DISP=OLD is generated when the 'QUIESCE HELD' flag does not exist for the database data set.

There are two new keywords for the GENJCL.IC command. They are used to override the defaults for the DISP= parameter. The GENJCL.IC keyword, DBQUIH, tells DBRC that the database will be quiesced when the image copy utility job is executed and it should use DISP=SHR. The keyword, NODBQUIH, tells DBRC that the database will not be quiesced when the utility is executed and DBRC should use DISP=OLD in the generated JCL job.

If the user already has canned GENJCL.IC JCL that is used for clean image copies, it may need to be changed to DISP=SHR for use with the database quiesce function.

IMS Version 11

DBRC CHANGE.DB and CHANGE.DBDS Commands

- CHANGE.DB DBD(dbdname) or CHANGE.DBDS AREA(dbdname)

  Keywords: `DBQUIH | NODBQUIH | DBQUI | NODBQUI`

  – DBQUIH
    – Set the QUIESCE HELD flag ON for specified database or area
  – NODBQUIH
    • Set the QUIESCE HELD flag OFF for specified database or area
  – DBQUI
    • Set the QUIESCE IN PROGRESS flag for specified database or area
  – NODBQUI
    • Set the QUIESCE IN PROGRESS flag off for specified database or area
- Keywords are typically used to correct DBRC RECON data set
  • Example:
    – Clean up the DBRC RECON data set at remote site for disaster recovery

21

The existing DBRC CHANGE.DB command has been enhanced to support the DB Quiesce function. The new keywords are NODBQUIH, DBQUIH, NODBQUI and DBQUI.

When the NODBQUIH keyword is specified on the CHANGE.DB or CHANGE.DBDS AREA(…) commands, the QUIESCE HELD flag for the specified database is set OFF. When the keyword is used for a Fast Path DEDB, the flag is reset for all of the areas of the DEDB. When the command is issued for a HALDB master, the flag is reset for all partitions of the HALDB database.

When the DBQUIH keyword is specified on the CHANGE.DB or CHANGE.DBDS AREA(…) commands, the QUIESCE HELD flag for the specified database is set ON. When the keyword is used for a Fast Path DEDB, the flag is set on for all of the areas of the DEDB. When the command is issued for a HALDB master, the flag is set for all partitions of the HALDB database.

When the NODBQUI keyword is specified on the CHANGE.DB or CHANGE.DBDS AREA(…) commands, the QUIESCE IN PROGRESS flag for the specified database is set OFF. When the keyword is used for a Fast Path DEDB, the flag is reset for all of the areas of the DEDB. When the command is issued for a HALDB master, the flag is reset for all partitions of the HALDB database.

When the DBQUI keyword is specified on the CHANGE.DB or CHANGE.DBDS AREA(…) commands, the QUIESCE IN PROGRESS flag for the specified database is set ON. When the keyword is used for a Fast Path DEDB, the flag is set on for all of the areas of the DEDB. When the command is issued for a HALDB master, the flag is set for all partitions of the HALDB database.

These commands should be used to correct error situations. For instance, they could be used to clean up the DBRC RECON data set at the remote site for disaster recovery purposes.

DBRC NOTIFY.ALLOC Command

- NOTIFY.ALLOC DEALTIME(…) QUIESCE
  – ALLOC record with deallocation timestamp caused by a DB QUIESCE

- Note:
  – Command used to correct DBRC RECON data set
    - Example:
      – Clean up the DBRC RECON data set at remote site for disaster recovery

22

The existing DBRC NOTIFY.ALLOC command has been enhanced to support the quiesce function. A new keyword, QUIESCE, has been added to the ALLOC record with a DEALTIME timestamp to indicate the ALLOC record was created by a quiesce command.

This commands should be used to correct error situations. For instance, it could be used to restore settings in the DBRC RECON data set at the remote site for disaster recovery purposes.

SELECTING ALLOCATION RECORDS

- %SELECT ALLOC
  - New Keywords
    - %ALLDBQ — Set to QUIESCE if the quiesce flag (ALLDBQUI) is on in the allocation record. Otherwise set to blanks.
    - %DBQSEL — Set to YES if the quiesce flag (ALLDBQUI) is on in the allocation record. Otherwise set to NO.
    - %DBQSELA — Set to YES if the quiesce flag (ALLDBQUI) is on in all allocation records selected. Otherwise set to NO.

23

This is an example of a %SELECT record that obtains database quiesce information

```
%SELECT ALLOC((POHIDJA,POHIDJAA),LAST)
    DBNAME              %DBNAME
    DDNAME              %DBDDN
    ALLOC TIME          %ALLTIME
    DEALL TIME          %DALTIME
    PRILOG TIME         %PLGTIME
    DBQUIESCE           %ALLDBQ
    DB QUIESCE FLAG     %DBQSEL
    ALL QUIESCE         %DBQSELA
%ENDSEL
```

An example of the output for this example is:

```
    DBNAME            POHIDJA
    DDNAME            POHIDJAA
    ALLOC TIME        08065235849764797-0800
    DEALL TIME        08066235849764797-0800
    PRILOG TIME       08063235849764797-0800
    DBQUIESCE         QUIESCE
    DB QUIESCE FLAG   YES
    ALL QUIESCE       YES
```

DBRC API: Output Blocks: TYPE=DB, TYPE=PART, TYPE=DBDS

Block	Flag	Field	Description
DSPAPQDB	'apqdb_AUFLAG'	apqdb_DBQui	QUIESCE IN PROGRESS
		apqdb_DBQuiH	QUIESCE HELD
		apqdb_DBQuiCmd	HALDB DEDB ON QUIESCE COMMAND
DSPAPQHP	'apqhp_AUFLAG'	apqhp_DBQui	QUIESCE IN PROGRESS
		apqhp_DBQuiH	QUIESCE HELD
		apqhp_DBQuiCmd	HALDB DEDB ON QUIESCE COMMAND
DSPAPQAR	'apqar_AUFLAG'	apqar_DBQui	QUIESCE IN PROGRESS
		apqar_DBQuiH	QUIESCE HELD
		apqar_DBQuiCmd	HALDB DEDB ON QUIESCE COMMAND
DSPAPQAL	'apqal_flags'	Apqal_DBQUI	QUIESCE Caused Deallocation

24

The DBRC Application Programming Interface (API) has been enhanced to provide information on the quiesce command status in the DBRC RECON data set. There are new flags returned by the DBRC Query request. The Query request returns information from the DBRC RECON data set for one or more registered databases. The allocation record will also return a new flag indicating if the deallocation time was added into the DBRC RECON data set as a result of a quiesce command. These new blocks will be returned for the following query types: 1) TYPE=DB, 2) TYPE=PART and 3) TYPE=DBDS.

XRF and RSR

- XRF
  - Alternate does not track quiesce command
    - Quiesce status is maintained across XRF takeover
      - Call to DBRC to obtain quiesce status
- RSR
  - Alternate will not track quiesce command
    - Quiesce status is not maintained across RSR takeover
  - DBRC RECON on alternate will not show quiesce status

25

The IMS Extended Restart Facility (XRF) alternate system will not track the quiesce command. However, the database quiesce status is maintained across an XRF takeover because an internal call is made to DBRC to obtain the status during the XRF takeover.

The Remote Site Recovery (RSR) feature in IMS will also not track the quiesce command status with the RSR alternate, and in the event of a remote site takeover, the fact that a database or area was quiesced at the active site will be unknown at the remote site.

Benefits

- Database Quiesce
  - Creates a coordinated recovery point across the IMSplex
  - RP is created without closing database data sets
    - No /DBR or /DBD
  - Transaction programs and BMPs are not terminated
    - Transaction and BMP update activity is quiesced after the next application synch point
  - Eliminates database outage when creating recovery points
    - Without DB quiesce outage typically takes several minutes
    - DB quiesce delay typically lasts only a few seconds

26

The benefits of the quiesce command are that it allows a recovery point to be created across the IMSplex without closing the database data sets. IMS quiesces transaction and BMP update activity after the next appplication checkpoint to provide a non-disruptive recovery point function.

Without the database quiesce function the creation of a recovery point typically takes several minutes. This is a database outage. With database quiesce function it typically takes only a few seconds to create a recovery point and there is no database outage. Instead, there is only a slight delay for some transactions and BMPs.

HALDB Online Reorganization

HALDB Online Reorg Performance Enhancements

- Invoke sequential access for VSAM KSDS get processing
  - Reduces CPU and elapsed time
- Eliminate GNP Call for Root-only DB
  - Reduces CPU and elapsed time
- Reduce use of the data set busy (ZID) lock
  - Updates for PHIDAM primary index are saved for insertion at the end of each unit of reorganization (UOR) so that the ZID lock is requested once per UOR.
  - Reduces CPU and elapsed time
- Eliminate block locks for ILDS updates
  - Reduces CPU and elapsed time

28

The performance of HALDB Online Reorganization is improved in six ways.

OLR VSAM KSDS Sequential Access

OLR is enhanced to take advantage of the VSAM sequential access option when issuing KSDS GET requests to retrieve sequentially from the input data set(s). This results in reduction in CPU and elapsed time.

Eliminate GNP Call for Root-only DB

For a root-only database, there is no reason to issue the GNP call since there is no dependent segment to be read. Eliminating the GNP call saves CPU and elapsed time.

Reduce use of the data set busy (ZID) lock during OLR

For a PHIDAM partition undergoing reorganization by OLR the updates for the primary index (KSDS) can be saved for insertion at the end of the unit of reorganization (UOR). By saving all the KSDS updates until the end of the UOR, the usage of the ZID lock for the primary index can be UOR changed. The change will be to obtain the ZID once before starting to insert all the saved primary index updates, and then released once. When the UOR covers many roots, many ZID lock requests will be eliminated which will result in a reduction of CPU usage and elapsed time.

Eliminate block locks for ILDS updates

For a HALDB partition with logical relationships or secondary indexes the updates for the indirect list dataset (ILDS) by OLR do not need to obtain the block (BID) lock. The BID lock is used for serialization of the updates to a block across IMS subsystems. The design of HALDB doesn't allow the ILDS to be updated by more than one IMS at a time since the ILDS is updated only by reorganizations and reorganizations for a partition are serialized.

HALDB Online Reorg Performance Enhancements

- Consolidate multiple updates for a block into one log record
  - Reduces logging, CPU, and elapsed time
- Implement IRLM lock look-aside
  - Avoids requesting a lock which is already held
  - Reduces CPU and elapsed time
- Benefits
  - Reduced CPU consumption
  - Shortened elapsed times
  - Reduced logging

29

Log reduction

For a HALDB partition undergoing OLR, the database update log records (type '50'x) will be consolidated when possible into full block updates. By combining all the updates for a full block into a single type '50'x log record, many of the small type '50'x log records will be eliminated. This will result in a reduction in the log volume generated by OLR. This can be significant. Typically, OLR has created three log records for each segment. There is one for the segment, one for the update of the pointer to the segment, and one for the change in free space. Since each type x'50' log record has a prefix of approximately 200 bytes, each segment generates log records of approximately 600 bytes plus the size of the segment. The reduction in the number of log records will significantly reduce the logging volume.

Lock reduction

Provide a lock look-aside function to reduce the number of times a lock request goes to the IRLM. The OLR process issues many lock requests to the IRLM. Many of these locks are already owned. By providing an IMS managed look-aside table for these locks, we can reduce the number of calls to the IRLM.

QUERY OLREORG and /DIS DB OLR

IMS 9 and IMS 10 SPEs

IMS 10 APARs: PK54615, PK54616;
IMS 9 APARs: PK36909, PK43203

- New statistics are shown
  - Segments moved, roots moved, option, status, and start time
- Statistics are retained across a termination and restart of OLR
  - QRY OLREORG and /DIS DB OLR after a restart of OLR will show the total bytes, roots, and segments moved by the OLR
- Statistics are available in RECON for suspended OLR
  - OLR is suspended by TERM or /TERM command
  - LIST.DB for partition database record returns the statistics
- Benefits
  - Easier to track the progress of OLR

30

SPEs for IMS 9 and IMS 10 added data returned by the QRY OLREORG and /DIS DB OLR commands. The SPEs for IMS 10 are APARs PK54615 and PK54616 (PTFs PK39696 and PK39697). The SPEs for IMS 9 are APARs PK36909 and PK43203 (PTFs PK32259 and either PK32354 for 900 or PK32260 for 901).

Previously, the number of bytes moved and the status were returned. Additional data now returned include the number of segments moved, the number of roots (database records) moved, the DEL or NODEL option, and the start time. In addition, three new status values may be returned. The new status values are:

RESUMED - This OLR has been resumed after being stopped for some reason such as a TERM OLREORG command or a user abend.
WAITRATE - This OLR is waiting due to the intentional delay because a value of less than 100 was specified on the RATE parameter.
WAITLOCK - This OLR is waiting for a lock.

The SPEs also preserve the bytes, segments, and roots moved values across the termination and restart of a OLR. When the OLR is suspended with a TERM or /TERM command and, then, restarted, the counts will include the data moved both before and after the suspension.

Additionally, the number of bytes moved, number of segments moved, and number of root segments moved statistics are available in the RECON partition database record between the time that the OLR is suspended (terminated) and the time it is restarted. An example of the addition to the RECON record listing is:

```
ONLINE REORG STATISTICS:
    OLR BYTES MOVED = 72156778
    OLR SEGMENTS MOVED = 423551
    OLR ROOT SEGMENTS MOVED = 72686
```

These enhancements are included in IMS 11.

QUERY OLREORG and /DIS DB OLR

IMS 9 and IMS 10 SPEs

QRY OLREORG example:

```
QRY OLREORG NAME(POHIDKA) SHOW(ALL)

Partition MbrName    CC LclStat  Rate  Bytes-Moved Segs-Moved
POHIDKA   IMS1        0 WAITRATE   50       334720       2310

Roots-Moved Option  Startt
        250 NODEL   2008.107 11:50:41.35
```

/DIS DB OLR example:

```
/DIS DB OLR

DATABASE PART     RATE       BYTES       SEGS    ROOTS STARTTIM
DBHDOJ01 PDHDOJD    50    72156778     423551    72686 08107/115049

STATUS
WAITRATE, OPTNODEL
```

31

This page shows examples of the commands. The first example is a QRY OLREORG command for partition POHIDKA which is currently waiting due to the use of the RATE parameter. The second example is a /DIS DB OLR command. The responses show the bytes moved, segments moved, and roots moved values, the start time, and status. In these examples the status is WAITRATE. for a restarted OLR. The restart time is also shown.

Additionally, the number of bytes moved, number of segments moved, and number of root segments moved statistics are available in the RECON partition database record between the time that the OLR is suspended (terminated) and the time it is restarted. An example of the addition to the RECON record listing is:

IRLM LOCKTIME Enhancement

Dynamic IRLM LOCKTIME

- IMS 9 and IMS 10 LOCKTIME parameter
  - Specified in DFSVSMxx member or DFSVSAMP DD
  - Used to timeout IRLM lock requests
    - U3310 abend or 'BD' status code issued
    - Separate values for MPP, IFP, and JMP vs. batch, BMP, and JBP timeouts

```
                    ABEND           ABEND
LOCKTIME=(mtime,STATUS,btime,STATUS)
```

- IMS 10 SPE and IMS 11 enhancements
  - Command to dynamically change LOCKTIME values
  - Command to show locktime values
- Benefits
  - IRLM lock timeouts may be controlled dynamically

IMS 10 APARS:
PK90139, PK90142

IMS 11 APARS:
PK82285, PK90134, PK90135

33

Maintenance for IMS 9 (APAR PK17824; PTF UK36902 for release 900 and PTF UK36903 for release 901) and IMS 10 (APR PK42705; PTF UK36904) have enhanced the capability to timeout lock requests when using the IRLM. Timeouts cause either a 'BD' status code or a U3310 abend of the requestor. The DFSVSAMP member or the DFSVSAMP DD data set is used to specify the options. The first pair of subparameters of the LOCKTIME parameter are used for MPP, IFP, and JMP regions as well as ODBA and CCTL (CICS) threads. The second pair are used to set the values for BMPs and JBPs.

The *mtime* subparameter is the only required one. It and *btime* are a number of seconds from 1 to 32767. If neither ABEND nor STATUS is supplied then it defaults to ABEND. If the *btime* value is supplied it will be used for batch, BMP, or JBP regions and the *mtime* value will then be used for other regions. If no *btime* time is supplied then the *mtime* time is used for all region types.

When the program is abended with U3310, the program and transaction are not stopped.

IMS 10 SPE and IMS 11 maintenance adds the capability to change the timeout values and to change the abend or status code action with a command. Additionally, this enhancement allows users to query the timeout settings. For IMS 10 the APAR numbers are PK90139 and PK90142. For IMS 11 the APAR numbers are PK82285, PK90134, and PK90135.

UPDATE Command for LOCKTIME

- Command

```
UPDATE IMS SET(LCLPARM
           (LOCKTIME(MSG(mtime),MSGOPT(ABEND|STATUS),
                    (BMP(btime),BMPOPT(ABEND|STATUS)))
```

  - Command may set timeout limit and option
    - MSG and MSGOPT: MPP, IFP, JMP, CCTL, or ODBA thread
    - BMP and BMPOPT: BMP or JBP

- Timeout values in IMS must be coordinated with IRLM TIMEOUT value
  - IRLM drives IMS based on its TIMEOUT value
    - IMS only checks its LOCKTIME value when it is driven by IRLM

34

The UPDATE IMS command has been enhanced to allow users to dynamically change the LOCKTIME values in online systems. MSG and MSGOPT refer to the time and option for MPP, IFP, and JMP regions and CCTL (DBCTL) and ODBA threads. BM{ and BMOPT refer to the time and option for BMP and JMP regions.

The LCLPARM parameter indicates that this command updates a local parameter. IMS 10 introduced the UPDATE IMS command with SET(PLEXPARM(…)). This is used to set global status indicators. IMS 11 extends the UPD IMS command to update local parameters. LOCKTIME is a local parameter in each IMS. The LOCKTIME values are updated in all of the IMS systems to which the command is routed.

The IRLM TIMEOUT value is used for two purposes. First, it controls the "long lock" value. When locks from this subsystem wait for longer than the TIMEOUT value, an SMF record is written and a message is issued. This can be used to analyze locking problems. This does not cause these lock requests to be rejected. Second, the IRLM TIMEOUT value determines when IMS will be informed of the wait for a lock. IMS only times out lock requests when IRLM informs it of these waits. If you set an IMS LOCKTIME of 30 seconds but the IRLM TIMEOUT value is 60 seconds, IRLM will only inform IMS of the wait after the lock request has waited 60 seconds. This situation is avoided adjusting the TIMEOUT value in IRLM. It may be changed with the F irlmproc,TIMEOUT=seconds,imsname command. The TIMEOUT value must be a multiple of the local deadlock parameter. If the value entered is not an even multiple of the local deadlock parameter, IRLM increases the timeout value to the next highest multiple.

QUERY Command for LOCKTIME

- Command

```
QUERY IMS TYPE(LCLPARM) SHOW(LOCAL|ALL)
```

- TSO SPOC Response

```
PLEX1   IMS Single Point of Control
Command ===>
---    Plex . .    Route . .    Wait . .
 Response for: QRY IMS TYPE(LCLPARM) SHOW(ALL)
MbrName CC CCText MSG MSGOPT    BMP BMPOPT
IMS2    00         10 STATUS     30 ABEND
IMS1    00         10 STATUS     60 ABEND
```

35

The QUERY IMS command has been enhanced to show the LOCKTIME values when TYPE(LCLPARM) is specified. When TYPE(LCLPARM) is specified the command is not routed to a command master. It retrieves information for the IMS systems to which the command is routed. When TYPE(PLEXPARM) is specified, one of the IMSs is a command master. It supplies values for the PLEX. Since there is no PLEXwide value for LOCKTIME, LCLPARM is used to retrieve LOCKTIME values.

The SHOW parameter for the QUERY IMS command may specify LOCAL, GLOBAL, or ALL. Since there are no GLOBAL values for the LOCKTIME parameter, LOCAL and ALL return the same data.

In this example, MPP, IFP, JMP, CCTL, and ODBA lock requests will time out after 10 seconds and result in a status code being returned for the call. BMP and JBP lock requests will time out in 30 seconds on IMS2 and in 60 seconds on IMS1. Time outs for BMPs and JBPs will result in abends.

As with previous releases, the IRLM TIMEOUT value may be queried with the F irlmproc,STATUS command. The timeout value for each IMS using the IRLM is shown in the T/OUT column.

Fast Path Enhancements

Fast Path Enhancements

- 64 Bit Buffer Manager
- OPEN option for UPD DB and UPD AREA commands
- Unique Subcode for PROCOPT=GO U1026 Abends
- Non-Recoverable DEDBs with SDEPs (SPE for IMS 10)
- Access Type by Area (SPE for IMS 9 and IMS 10)

Fast Path 64 Bit Buffer Manager

Fast Path 64 Bit Buffer Manager

- Fast Path buffers above the 2GB bar in control region address space
  - Optional
  - Multiple subpools with different buffer sizes
    - Subpools expand on demand
- Implementation
  - DFSDFxxx PROCLIB member

```
<SECTION=FASTPATH>
FPBP64=Y,FPBP64M=xxxxxxxxxx
```

- FPBP64=N is the default
- Changing FPBP64 value requires a cold start of IMS
- User does not specify the number of buffers
  - DBBF, BSIZ, and DBFX are ignored when FPBP64=Y is specified
- FPBP64M sets maximum storage used - this is explained later

39

IMS 11 provides an option to place the Fast Path buffers above the 2GB bar. When this option is used, buffer management is changed significantly. Instead of having a fixed number of buffers of a single size, there are multiple subpools. The buffers in the different subpools have different sizes. The subpools expand as the need for buffers increases.

The 64 bit Fast Path buffer manager is used when FPBP64=Y is specified in the FASTPATH section of the DFSDFxxx PROCLIB member. The FASTPATH section is new in IMS 11. FPBP64=N is the default. This means that the 64 bit buffer manager is only used if explicitly requested.

Changing from FPBP64=N to FPBP64=Y or from FPBP64=Y to FPBP64=N requires a cold start of IMS.

FDBR and XRF systems must have the same specification as the system which they are tracking.

The DBBF, BSIZ, and DBFX values are not used by the 64 bit buffer manager. Their specifications are ignored when FPBP64=Y is used.

The FPBP64M= parameter may be used to limit the storage above the bar which is used. This is explained later.

Fast Path 64 Bit Buffer Manager

- 64 bit buffer pool
  - 64 bit control region private storage or common storage
  - Initial allocation of buffers in each subpool is determined by the number of areas using the CI size
  - Subpools expand if dependent regions require more buffers
    - Maximum storage used above the bar is 2GB or FPBP64M limit
- ECSA pool
  - Every subpool has a small number of buffers in ECSA
    - Used for:
      - MSDBs
      - Buffer headers (DMHRs) and some control blocks
      - System buffers including those used for SDEP inserts and FLD calls
  - Initial allocation of ECSA storage is 64K (tracking of 2GB above the bar)
    - ECSA storage may grow as needed

The 64 bit buffer pool is allocated in private storage with z/OS V1R9 since this release of z/OS does not have common storage above 2GB. With z/OS V1R10 the pool is allocated in common storage. z/OS V1R10 introduced common storage above 2GB.

When the 64 bit buffer pool is initialized, the number of buffers in each subpool is determined by the number of areas using each CI size. The number of buffers in each subpool is then increased depending on the use of them by applications. The storage is allocated in control region private 64-bit storage.

The maximum amount of storage allocated in 64 bit storage for the DEDB buffers is limited to 2GB or the limit specified by the FPBP64M= parameter. 2GB will support over 262,000 8K buffers. It is unlikely that this limit will be reached by today's systems. The buffer manager only needs buffers to contain active data in currently running regions and threads. Nevertheless, if this limit is reached, non-message driven BMPs and JBPs will receive status code FW. They may use this as an indication that a sync point should be taken to release buffers. If this does not free enough storage to allow IMS to continue operation, UOWs for non-message driven BMPs and JBPs will be aborted and they will receive an FR status code. Aborting the UOW will free buffers in 64 bit storage. Only non-message driven BMPs and JBPs receive the FW and FR status codes. Other regions and threads are abended when they cannot get a buffer. This also occurs when they require more buffers than their NBA+OBA allocation.

Even though most buffers are above the 2GB bar, there are some buffers below the bar in ECSA. These buffers are used for MSDBs, buffer headers, some control blocks, and some system buffers. Buffer headers are the DHR control blocks. System buffers are used for SDEP inserts and FLD calls. The initial allocation of buffers in ECSA is 64K. This storage is associated with tracking the 2GB of storage above the bar. The storage in ECSA may grow as needed.

Maximum Size for Fast Path 64 Bit Buffer Pool

- FPBP64M parameter sets maximum size of the pool
  - DFSDFxxx PROCLIB member

```
<SECTION=FASTPATH>
FPBP64=Y,FPBP64M=xxxxxxxxxx
```

    - FPBP64M= must be specified (no default)
    - Values from 1M to 2G-1 are valid

- UPDATE POOL(FPBP64) SET(LIMIT(xxxxx)
  - Command changes maximum pool size
  - LIMIT value must be between current pool size and 2G-1
  - LIMIT from the command is not restored across restarts
    - Change value in DFSDFxxx for permanent changes

41

You limit the maximum amount of storage used for the Fast Path 64-bit buffer pool with the FPBP64M= parameter. It must be specified in the the DFSDFxxx PROCLIB member when FPBP64=Y is used. Valid values for FPBP64M are 1M through 2G-1. Values may be specified in bytes, K, or M. For example FPBP64M=100000000, FPBP64M=150000K, and FPBP64M=180M are valid specifications.

The limit specified in DFSDFxxx may be changed with the UPDATE POOL(FPBP64) SET(LIMIT(xxxxx)) command where 'xxxxx' is the new value. The new value cannot be smaller than the amount of storage currently used and cannot be greater than 2G-1. The value specified in the command is not restored on a restart of IMS. Instead, the value in the DFSDFxxx member is used. If you want to make a permanent change in the value, you should also specify the new limit in the DFSDFxxx member.

Fast Path 64 Bit Buffer Manager Statistics

- Log statistics
  - x'5945' log record includes buffer use and buffer wait information for UOW
  - x'4516' system checkpoint log record includes buffer statistics
  - Fast Path Log Analysis utility (DBFULTA0) processes these log records
- QUERY POOL TYPE(FPBP64) SHOW(ALL) command
  - Shows information about 64 bit subpools
    - Includes
      - Numbers of buffers of each size
      - Expansions for each buffer size
      - Numbers of buffers in use
  - Example response follows.

The x'5945' log record contains statistics about the use of 64 bit FP buffers by a unit of work. This includes information about the use of buffers and waits for buffers. At system checkpoint time IMS logs a x'4516' log record with 64 bit FP buffer statistics. These log records are read and processed by the Fast Path Log Analysis utility (DBFULTA0).

The QUERY POOL command has been enhanced to show information about the 64 bit buffer manager. An example command response is shown on the next two pages.

There is one line for each subpool above the 2GB bar and one line for each subpool in ECSA.

QUERY POOL TYPE(FPBP64) reply - part 1 (left columns)

Subpool	MbrName	CC	Size	Type	Tot_Buf	Buf_Use	Buf_Ava	%Use
DBF_MAXB	SYS3							
DBF_TOTB	SYS3			G	880	8	872	
DBFC0001	SYS3		512	A	320		320	
DBFC0001	SYS3	0		B	80	0	80	0
DBFC0001	SYS3	0		E	40	0	40	0
DBFC0001	SYS3	0		E	40	0	40	0
DBFC0001	SYS3	0		E	40	0	40	0
DBFC0001	SYS3	0		E	40	0	40	0
DBFC0001	SYS3	0		E	40	0	40	0
DBFC0001	SYS3	0		E	40	0	40	0
DBFC0002	SYS3		1024	A	80		80	
DBFC0002	SYS3	0		B	80	0	80	0
DBFC0003	SYS3		2048	A	80		80	
DBFC0003	SYS3	0		B	80	0	80	0
DBFC0004	SYS3		4096	A	80		80	
DBFC0004	SYS3	0		B	80	0	80	0
DBFS0001	SYS3		512	A	80	8	72	
DBFS0001	SYS3	0		B	80	8	72	10
DBFS0002	SYS3		1024	A	80		80	
DBFS0002	SYS3	0		B	80	0	80	0
DBFS0003	SYS3		2048	A	80		80	
DBFS0003	SYS3	0		B	80	0	80	0
DBFS0004	SYS3		4096	A	80		80	
DBFS0004	SYS3	0		B	80	0	80	0

43

This is an example response for the QRY POOL TYPE(FPBP64) SHOW(ALL) command. This page shows the first columns in the response. The next page shows the other columns.

The meanings of the columns shown here are:

subpool: The name of the subpool. This is the internal name of the pool where xxxx is a numeric value.

- DBFCxxxx A common subpool used for DEDB data. The buffers reside in 64-bit addressable storage.
- DBFXxxxx A system subpool used for all other buffer requests, including IMS internal buffers. The buffers reside in ECSA
- DBF_MAXB This is not a subpool. This line includes the FPBP64M value in the 64b_Buf column.
- DBF_TOTB This is not a subpool. This line includes the overall totals for the entire buffer pool.

MbrName: The IMS identifier. The same as in other type-2 command responses.

CC: Command completion code. The same as in other type-2 command responses.

Size: The buffer size

Type: Describes what this row describes.

- G: This row contains overall totals for the entire buffer pool.
- A: The total values for the subpool and extents with the name of the subpool in the SUBPOOL column.
- B: This is the base section of the subpool. It does not include the extent values.
- E: This is an extent for the subpool. It does not include the base section of the subpool.

Tot_Buf: The total number of buffers in this subpool or for this subtotal including the base section and the extents.

Buf_Use: The number of buffers being used by a process (IMS, dependent region or external thread (such as ODBM)) from this subpool, or extents or the grand total for the entire buffer pool.

Buf_Ava: The number of buffers available for use from this subpool or extents or the grand total for the entire buffer pool.

%Use: Percentage of buffers in use by a process for a subpool or extents.

QUERY POOL TYPE(FPBP64) reply - part 2 (right columns)

```
%Ext ECSA_Tot ECSA_Buf ECSA_Oth 64b_Tot 64b_Buf Date Created
                                  2047M
           1M     600K     389K    720K    720K
         246K                      160K
  50                        40K             40K 2009.041 17:03:40.23
                            17K             20K 2009.041 17:11:15.41
                            17K             20K 2009.041 17:11:14.31
                            17K             20K 2009.041 17:11:13.79
                            17K             20K 2009.041 17:11:13.74
                            17K             20K 2009.041 17:11:13.71
                            17K             20K 2009.041 17:11:13.70
          34K                      80K
  50                        34K             80K 2009.041 17:03:40.23
          34K                     160K
  50                        34K            160K 2009.041 17:03:40.23
          34K                     320K
  50                        34K            320K 2009.041 17:03:40.23
          74K
  50                40K     34K                 2009.041 17:03:40.23
         114K
  50                80K     34K                 2009.041 17:03:40.23
         194K
  50               160K     34K                 2009.041 17:03:40.23
         354K
  50               320K     34K                 2009.041 17:03:40.23
```

44

The meanings of the columns shown here are:

%Ext: Percentage of the base section to be utilized before and extent is taken. This value might change over time based on buffer usage.

ECSA_Tot: The total amount of ECSA used for each subpool and extents, and the overall ECSA total.

ECSA_Buf: The amount of ECSA used for each subpool and for each extent.

ECSA_Oth: The amount of ECSA allocated for control blocks.

64b_Tot: The total amount of 64 bit storage used for each subpool and extents, and the overall 64 bit storage total.

64b_Buf: The 64 bit storage used for buffers for each subpool and extent.

Date created: The date and time this subpool or extent was created.

Fast Path 64 Bit Buffer Manager

- I/O is done directly from and to the 64 bit buffers
- OBA is not serialized
  - Each dependent region or thread may have its OBA allocation at the same time
    - Eliminates a potential delay or bottleneck
- DBFX is not applicable
  - Buffers are created and fixed as needed

I/Os are done directly for the buffers in 64 bit storage (above the 2GB bar). CIs do not have to be moved to "below the bar" buffers to do I/O.

When the 64 bit buffer manager is used, the use of OBA buffers is not serialized. Multiple dependent regions or threads may be using their OBA allocations at the same time. This eliminates a potential bottleneck in buffer use. In practice it means that each region or thread may be using the number of buffers equal to its NBA + OBA specification.

When the non-64 bit buffer manager is used, DBFX specifies the number of buffers that are page fixed in addition to those calculated from NBA and OBA specifications. This is done to ensure that there are page fixed buffers for the use of output threads. This concept does not exist with the 64 bit buffer manager. The user does not have to specify a DBFX value.

Fast Path 64 Bit Buffer Manager

- Parallel DL/I considerations
  - z/OS V1R9 does not have 64-bit common storage
    - Private storage is used
    - Parallel DL/I is required
  - z/OS V1R10 has common and private 64-bit storage
    - Common storage is used
    - Parallel DL/I is not required, however, it is recommended
      - PARDLI=1 is honored

All 64 bit storage with z/OS 1.9 is private. There is no common 64 bit storage. This means that the buffers must be in private storage of the Control Region. This means that parallel DL/I is always used with z/OS 1.9 when using the 64 bit buffer manager. That is, you cannot disable parallel DL/I by specifying PARDLI=1 for a dependent region.

z/OS 1.10 adds support for 64 bit common storage. This is used for the Fast Path 64 bit buffers. With z/OS 1.10 parallel DL/I is optional, but still recommended. The default for the PARDLI parameter remains PARDLI=0 which invokes parallel DL/I. To turn off parallel DLI, one must specify PARDLI=1.

There is some performance penalty for the use of the 64 bit buffer manager with z/OS 1.9. This is due to referencing storage in another address space. On the other hand, the overhead is typically insignificant when compared to the path length of an application transaction.

Fast Path 64 Bit Buffer Manager

- Benefits
  - ECSA constraint relief
    - Eliminates U1011 abends due to ECSA fragmentation
  - Self tuning
    - User does not specify the number of buffers
    - System dynamically adds buffers when required
  - Supports multiple buffer sizes
    - Better use of buffers when using areas with different CI sizes
  - IMS restart is not required to add more Fast Path buffers
    - New dependent regions or threads may be added
      - New regions or threads may have higher NBA and/or OBA values

47

The use of the Fast Path 64 bit buffer manager provides several benefits.

Obviously, it provides ECSA constraint relief. This could eliminate some U1011 abends. U1011 abends can occur during control region initialization when Fast Path initialization cannot get storage for its buffer pool.

The 64 bit buffer manager is self tuning. Users do not have to calculate the number of buffers required. Instead, the buffer manager reacts to the requirements of the system by creating the number of buffers required.

The 64 bit buffer manager makes better use of storage. It can create buffers of different sizes. This makes the use of DEDBs with different CI sizes more efficient. This can be especially important for users who want to merge IMS systems which in the past have had DEDBs with different CI sizes.

As systems grow, more regions or application threads may be required. As applications change, regions and threads may require more buffers. The number of buffers in the system must grow as regions or threads are added or as their NBA and OBA values are increased. With the old Fast Path buffer manager adding buffers required a restart of IMS. This is not required with the 64 bit buffer manager. Instead, buffers are added by the system as they are needed.

Fast Path Usability and Serviceability

48

OPEN Option for UPD DB and UPD AREA command

- OPEN Option on UPD DB and UPD AREA commands for DEDBs
  - Used with UPD … START(ACCESS) command
  - Opens all areas of specified DEDB or the specified AREA
  - Examples:
    - UPD DB NAME(DEDB001) START(ACCESS) OPTION(OPEN)
      - Opens all areas for DEDB001
    - UPD AREA NAME(AREA0102) START(ACCESS) OPTION(OPEN)
      - Opens area AREA0102
- Operational consideration
  - OPEN option is not logged
    - Warm start of IMS will not open these areas
- Benefit
  - Operations may open areas before their first use by applications

49

The OPTION(OPEN) parameter is added to the UPD DB START(ACCESS) and UPD AREA START(ACCESS) commands for DEDBs. This causes the areas to be opened by the UPD command.

Since the open option is not logged, a restart of IMS or a reconnect to the IRLM will not open these areas unless (1) the areas are defined as PREOPEN, (2) FPOPN=R is specified for emergency restarts, or (3) FPRLM=R is specified for IRLM reconnects. In other words, a normal termination of IMS followed by a warm restart will not open the areas unless they are defined as PREOPEN.

OPTION(OPEN) allows operations to open area data sets before they are first used by applications. This eliminates a potential delay for the application as it waits for the open to complete.

Unique Subcode for PROCOPT=GO U1026 Abend

- New Fast Path abend subcode for PROCOPT=GO "pointer errors"
  - U1026 subcode x'5A' used with PROCOPT=GO
  - Previous releases returned subcode x'53' for both:
    - Actual pointer errors using PROCOPT=G, I, R, D, or A
    - Apparent pointer errors using PROCOPT=GO
- Benefits
  - Lengthy analysis is not required to determine that an actual pointer error has not occurred

50

U1026 abends are used for many Fath Path problems. These abends include a subcode which indicates the type of problem encountered.

When accessing DEDBs some users have needed a lengthy diagnosis for U1026 abends with subcode x'53'. Previous releases of IMS returned the an x'53' subcode when either an actual pointer error was encountered when using PROCOPT=G, I, R, D, or A or an apparent pointer error was encountered when using PROCOPT=GO. PROCOPT=GO is "read without integrity". Calls using PROCOPT=GO may encounter apparent pointer errors which are not actual pointer errors because locking is not used to ensure integrity. If PROCOPT=GO is used, one may reasonably assume that an "apparent pointer error" is not an indication of an actual problem. On the other hand, if PROCOPT=G is used, an x'53' subcode indicates that an actual problem exists. Determining if PROCOPT=G or PROCOPT=GO was used for the call may require a lengthy analysis with previous releases of IMS.

IMS 11 has changed this processing when using PROCOPT=GO. The x'53' subcode is no longer returned when using PROCOPT=GO. Instead, the new subcode x'5A' is returned. This allows users to avoid a lengthy analysis.

Non-recoverable DEDBs with SDEPs (IMS 10 SPE)

IMS 10 APARs: PK56321; PK65219

- DEDBs with SDEPs may be made non-recoverable
  - Previously, DEDBs with SDEPs could not be defined as non-recoverable
  - DBRC Command:
    - INIT.DB … NONRECOV
    - CHANGE.DB … NONRECOV
- Operational considerations
  - Data may be lost across IMS failure and restart
    - Log record x'5951' does not contain changed data
  - DFS3711W issued when changed data is missing from log record on restart
  - SDEP SCAN may be used to read existing data
  - SDEP DELETE may be used to clean out SDEP data
- Benefit
  - More efficient way to write data that is not critical

51

This enhancement has been delivered for IMS 10 with APARs PK56321 and PK65219 (PTF UK35862). It is included in IMS 11.

The text of the DFS3711W message is:

DFS3711W NONRECOVERABLE DEDB INTEGRITY WARNING DEDB dedbname AREA areaname

This is the same message that is issued when other potential integrity problems are found with non-recoverable DEDBs.

One user plans to implement non-recoverable DEBDs with SDEPs for use with an application trace. This application has implemented a trace which is written to SDEPs by the application. The use of a non-recoverable DEDB will minimize the logging done for the trace. The use of SDEPs will minimize the overhead of the trace. This trace is not a critical business process. In the event of an IMS failure, the loss of the trace in the SDEPs would be acceptable.

Access Type by Area (SPE for IMS 9 and IMS 10)

IMS 9 APAR: PK65582; IMS 10 APAR PK74403; IMS 11 APAR PK78042

- Access Type may be set for individual areas
  - Previously, access type was set only for DEDBs
- New ACCTYPE parameter for SET on UPD AREA command

```
UPD AREA NAME(areaname) START(ACCESS) SET(ACCTYPE(accesstype))
```

  - 'accesstype' may be BRWS, READ, UPD, or EXCL
  - 'accesstype' may not be set higher than that of the DEDB
    - e.g. if DEDB is READ, its areas cannot be UPD
    - If DEDB access type is changed and area value is higher, the DEDB value overrides the area value

52

An enhancement has been made to allow an UPD AREA command to set the access type value for an area to a value other than that for the DEDB. Previously, all areas of a DEDB had the same access type value as the DEDB. This enhancement is available in IMS Version 9 (APAR PK65582; PTF UK38341), Version 10 (APAR PK74403; PTF UK42764) and Version 11 (APAP PK78042; PTF UK42765).

The enhancement allows one to specify SET(ACCTYPE(value)) in an UPDATE AREA NAME(areaname) START(ACCESS) command. The value must be no higher than that for the area's DEDB. For example, if the DEDB access type is READ, its areas cannot have their access types set to UPD.

If the DEDB's access type value is set to a value lower than that for any of its areas, the areas' access type values are changed to that for the DEDB. DEDB access type values may be set with either an UPD DB NAME(dbname) START(ACCESS) SET(ACCTYPE(value)) command or a /START DB dbname ACCESS=value command.

When the access type value is changed from UPD or EXCL to BRWS or READ, the area is closed and opened for read. Similarly, when the access type value is changed from BRWS or READ to UPD or EXCL the area is closed and opened for update.

The area access type value is restored across an IMS warm or emergency restart.

IMS Version 11

Access Type by Area (SPE for IMS 9 and IMS 10)

- QUERY AREA and QUERY DB enhanced
  - Show access for AREA when it differs from that of the database
- Access conflict
  - If area access type is lower than that required by call, 'FH' status code is returned
- Benefits
  - Areas may be handled separately for database maintenance operations
    - Area may be set to ACCESS=READ for clean image copy while others are available for update
  - Areas may be handled separately for special application processes
    - Area may be set to ACCESS=BRWS in one subsystem while another updates it

53

A QUERY DB or QUERY AREA command will show the access type value for individual areas if they differ from that of the database. ACCTYPE can now be specified as a filter on the QUERY AREA SHOW(filtername) command.

A status code of FH is returned when the area access type value is lower than that required by the DLI call. For example, if the access type value is READ, an ISRT call would receive the FH status code. As in previous releases, the FH status code is also returned when the area has the same access type value as the database and the area is stopped.

Fast Path Enhancements

- 64 Bit Buffer Manager
  - Self tuning buffer manager
- OPEN option for UPD DB and UPD AREA commands
  - Command to open DEDBs and AREAs
- Unique Subcode for PROCOPT=GO U1026 Abends
  - Easier determination if actual pointer error exists
- Non-Recoverable DEDBs with SDEPs (SPE for IMS 10)
  - Lower overhead for creating non-critical data
- Access Type by Area (SPE for IMS 9 and IMS 10)
  - Individual areas may have their own access type value

Database Enhancements

- Database Quiesce
  - Creation of recovery points with minimal disruption
- HALDB Online Reorganization enhancements
  - Performance improvements
  - Statistics and restart enhancements
- IRLM Locktime enhancement
  - Dynamic changing of lock timeout values
- Fast Path

55

IMS Version 11

Transaction Manager

Information Management software

Transaction Manager

- Type-2 Query Commands
- DFSMSCE0 Enhancements
- Full Function Response Mode Recovery
- Shared Queues False Scheduling Reduction
- Transaction Expiration

Type-2 QUERY Commands for TM Resources

Overview

- Prior to IMS 11, attributes/status could be displayed using the type-2 QUERY command for all IMS DB resources, but only some IMS TM resources:
  - Transactions
  - MSC definitions
- In IMS 11, type-2 QUERY command enhanced to support all IMS TM resources
  - LTERMs
  - NODEs
  - USERs
  - USERIDs
- Display IMS TM resource attributes/status by issuing the QUERY command through any OM API such as TSO SPOC, Manage Resources or to an entire IMSplex using the Batch SPOC Utility
- Benefits
  - Improved ease-of-use for managing IMS TM resources
  - Increase efficiency by consolidating output of several type-1 commands into a single type-2 QUERY command

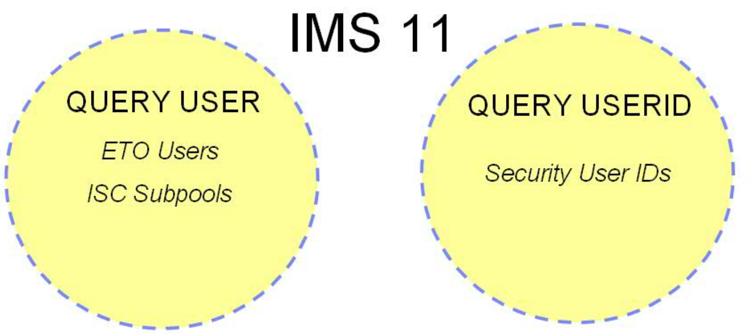
4

IMS 11 introduces the new capability of querying all TM resources. Prior to IMS 11, only DB resources and a subset of TM resources could be queried. The TM resources that now have type-2 QUERY command support in IMS 11 are LTERMs, NODEs, USERs, and USERIDs.

With this enhancement, IMS TM resources are easier to manage and additionally, output from several Type-1 commands is now consolidated into a single Type-2 command's output.

Querying IMS TM Resources

- QUERY LTERM – query static and ETO logical terminals (LTERMs)
- QUERY NODE – query VTAM terminals/nodes, and non-VTAM devices (system console, SPOOL, TCO)
- QUERY USER – query ETO users, and ISC subpools
- QUERY USERID – query user IDs (security RACF user IDs)



5

Here is a little more detail about the IMS TM resources that you can query using the type-2 QUERY command. First we have QUERY LTERM. You can use this command to display the attributes and status of both static and ETO logical terminals. This command can also be issued on an XRF-alternate IMS but only information local to that IMS will be displayed (in other words, there is no global support when QUERY is issued on an XRF-alternate IMS). A static terminal is a terminal that was created by the IMS SYSGEN process, whereas an ETO logical terminal is a terminal that has been created dynamically.

Next we have QUERY NODE. You can use this command to display information about VTAM terminals represented by nodes, as well as non-VTAM devices (like the system console and SPOOL and SYSOUT devices) across the IMSplex. This command can be specified only through the OM API and is valid on an XRF alternate.

There is also QUERY USER and QUERY USERID. Let’s clarify the difference between these two. A USER refers to either a dynamic/ETO user or an ISC subpool user (which can be either static or dynamic/ETO). On the other hand, a USERID refers to the IDs that are associated with RACF security. We will see examples of these throughout the remaining foils.

Prior to IMS 11, you could issue the type-1 /DISPLAY USER command to display ETO users, ISC subpools, and security user IDs. With the enhancement to the QUERY command in IMS 11, you can now 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.

QUERY LTERM Command Syntax

```
QUERY LTERM NAME(ltermname1, ltermname2,...)
MSGAGE(number)
QCNT(qualifier,number)
OPTION(MSGQ | EMHQ)
SHOW(LOCAL | GLOBAL | attribute(s))
STATUS(status)
```

- NAME() = one or more LTERM names, wildcard supported
- MSGAGE() = filter for only displaying LTERMs that have messages of a minimum age (0-365 valid) on the shared queue
- QCNT() = filter for only displaying LTERMs with a specified queue count
  - LT | LE | GT | GE | EQ | NE + number
- OPTION() = indicates queue to get count from
  - Can only specify this parameter with MSGAGE or QCNT filter included

6

Let's look at the syntax for the QUERY LTERM command. The syntax is "QUERY LTERM" followed by some parameters.

First is the MSGAGE parameter, where you specify a value indicating the minimum age of the LTERM messages in the shared queue that should be queried. So for example, if you specified a value of MSGAGE(3), you'd see the LTERMs that have queued messages either three or more days old in the command output.

Next is the QCNT parameter, which allows you to specify a qualifier pertaining to the LTERMs' queue counts. You can specify any of the following qualifiers followed by a numerical value in order to selectively display certain LTERMs based on queue count:

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

Next is the OPTION parameter, where you indicate where the queue count data should be drawn from. You can specify OPTION(MSGQ) if you're using shared queues and the queue count should be gotten from the shared queue. Alternatively, you can specify OPTION(EMHQ) if you'd like to display the queue count from expedited message handler queue.

The gold stars shown in this and the next several command syntax visuals represent filters that allow you to selectively display resources that have the designated filter values.

QUERY LTERM Command Syntax

```
QUERY LTERM NAME(ltermname1, ltermname2,...)
MSGAGE(number)
QCNT(qualifier,number)
OPTION(MSGQ | EMHQ)
SHOW(LOCAL | GLOBAL | attribute(s))
STATUS(status)
```

- SHOW() = display local/global resource queue counts or other attribute info
- STATUS() = displays only the LTERMs that have specified status(es)

7

Next is the SHOW parameter, where you can indicate whether you'd like to display local LTERMs, global LTERMs or specific attributes of the LTERM. Lastly, the STATUS() parameter is used as a filter to display only LTERMs that have a specific designated status.

Additional Notes

The behavior of the new type-2 QUERY commands for TM resources is in contrast to existing type-1 commands.

With type-1 /DISPLAY commands, IMS attempts to determine where the resource (node, user, lterm) is active. If a resource is active (owned) on a particular IMS, then the owning IMS displays the actual global status. All other IMS systems (including the master if it is not the owning system) display local resource status only. For diagnostic purposes, this can cause a problem if local status does not match global status (which may indicate a problem in the resource structure), in that it makes it more difficult to diagnose potential problems.

With type-2 query commands, the command master is the only system that displays global status from the resource structure, regardless of whether a resource is active on a particular system. In addition, every system (including the master) displays status local to that system. While this can cause duplicate information to be displayed (the command master would normally display the same global and local status), it greatly improves the ability to diagnose potential problems that might exist.

QUERY LTERM

SHOW Parameter	Meaning
COMPONENT	Input/Output Component Numbers
EMHQ	Shared EMHQ message queue count
MSGAGE	Message age (shared message queues)
MSNAME	Associated MSNAME (for remote LTERMs)
NODE	Associated node
OWNER	Owner (IMSID in resource structure)
QCNT	Full-function message queue count
STATUS	Status of LTERM
USER	Associated user
VERSION	Version number (in resource structure)

For reference, here is a table listing each of the possible SHOW parameter values along with their meanings for the QUERY LTERM command.

QUERY NODE Command Syntax

```
QUERY NODE NAME(nodename1,nodename2,...)
USER()SHOW(LOCAL | GLOBAL | attributes)
STATUS(status)
```

- NAME() = one or more VTAM nodes (terminals) or non-VTAM devices that are to be displayed, wildcards supported
  - Non-VTAM devices: specify a name of DFSLN*xxx*, where *xxx* is the LINE number of the device
    - System Console (always LINE 1, for example DFSLN001)
    - SYSOUT (DISK, PUNCH, PRINTER, READER, TAPE)
    - SPOOL
    - TCO

9

Here is the command syntax for QUERY NODE. On the NAME() parameter, one or more VTAM node names/IMS terminals or non-VTAM devices can be listed. There is wildcard character support for the NAME() parameter. In the case of non-VTAM devices, the standard way of specifying the names is to reference them with a name beginning with DFSLN, followed by a line number for the device. An example is DFSLN001 to represent the system console.

The types of non-VTAM devices that can be referenced on the NAME() parameter are listed here in this visual.

QUERY NODE Command Syntax

```
QUERY NODE NAME(nodename1,nodename2,...)
USER(username1,username2,...)
SHOW(LOCAL | GLOBAL | attributes)
STATUS(status)
```

- USER() = filter for displaying only the ISC user name(s) allocated to node, wildcard supported
- SHOW() = display local/global resource queue counts or other attribute info
- STATUS() = filter for displaying only resources with specified status(es)

10

The next parameter is the USER() filter where the ISC user name(s) allocated to the node is specified. A wildcard character can be used for this parameter value as well.

The SHOW() parameter 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.

Lastly, the STATUS() parameter is used as a filter to display only NODEs that have a specific designated status.

QUERY NODE

SHOW Parameter	Meaning
AFFIN	IMS APPLID which owns VGR affinity
CID	VTAM Connection Identifier
CONV	IMS Conversation ID, transaction, and status
COUNT	Number of message sent and received
EMHQ	Shared EMHQ message queue count
ID	Other half-session qualifier for ISC parallel session
LTERM	Associated LTERMs
MODETBL	Mode table name associated with node
OWNER	Owner (IMSID Status of NODE
PRESET	Preset destination name
QCNT	Full-function message queue count
RECOVERY	Status Recovery Mode

11

For reference, here is a table listing each of the possible SHOW parameter values along with their meanings for the QUERY NODE command.

QUERY NODE

SHOW Parameter	Meaning
STATUS	Returns local or global status of the node
TYPE	Node type (FIN, SLUP, SLU1, SLU2, etc)
USER	Associated user
USERID	Signed-on user ID
VERSION	Version number (in resource structure)

This is a continuation of the list.

QUERY USER Command Syntax

```
QUERY USER NAME(username1,username2,...)
SHOW(LOCAL | GLOBAL | attributes)
STATUS(status)
```

- NAME() = one or more dynamic/ISC subpool users that are to be displayed, wildcards supported
- SHOW() = display local/global resource queue counts or other attribute info
- STATUS() = filter for only displaying resources with specified status(es)

For the QUERY USER command, the NAME() parameter is used to list the ETO, or dynamic users as well as the ISC subpool users that should be displayed in the command output. A wildcard character can be specified for this parameter as well. Much like the QUERY command for the other TM resources, the SHOW() parameter 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. The status filter is used to display USERs with a specific status.

QUERY USER

SHOW Parameter	Meaning
AUTOLOGON	Autologon information
CONV	IMS Conversation information (Conversation ID, transaction, and status)
EMHQ	Shared EMHQ message queue count
ID	For an ISC parallel session, the other half-session qualifier
LTERM	Associated LTERMs
NODE	Associated node
OWNER	Owner (IMSID in resource structure)
PRESET	Preset destination name
QCNT	Full-function message queue count
RECOVERY	Status Recovery Mode
STATUS	Status of USER
USERID	Signed-on user ID
VERSION	Version number (in resource structure)

For reference, here is a table listing each of the possible SHOW parameter values along with their meanings for the QUERY USER command.

QUERY USERID Command Syntax

```
QUERY USERID NAME(useridname1,useridname2,...)
SHOW(LOCAL | GLOBAL | attributes)
STATUS(status)
```

- NAME() = one or more user IDs that are to be displayed, wildcards supported
- SHOW() = display local/global resource queue counts or other attribute info
- STATUS() = filter for only displaying resources with specified status(es)

15

Here is the syntax for the QUERY USERID command. NAME() can be one or more RACF User IDs that are to be displayed in the command output, including wildcard character support. Much like the other TM QUERY commands, SHOW can be used to display local/global USERID queue counts as well as specific attribute values. STATUS is a filter that can be used to display USERIDs with a certain designated status.

QUERY USERID

SHOW Parameter	Meaning
NODE	Associated node
OWNER	Owner (IMSID in resource structure)
STATUS	Status of USER
USER	Associated user
VERSION	Version number (in resource structure)

For reference, here is a table listing each of the possible SHOW parameter values along with their meanings for the QUERY USERID command.

Operational Considerations

- Type-2 Commands are only supported through the Operations Manager
- Command Master may show global and local resource information
- Non-command master systems show local information only

There are a few operational items to consider. To use these Type-2 QUERY commands, the only required component is Operations Manager. There will be one command master IMS that processes the command each time one is entered. If there is global information to be displayed as well as local information, the command master IMS will display its own local information on one line of output, and the global information on a separate line. All of the other IMSs that are not the command master would display their own local information only.

Performance Considerations

- No effect on mainline performance
- Wildcard parameters may cause significant access to the Resource Structure, the Shared Message Queues, or both
  - NAME(*) is the default
- MSGAGE() information requests may cause all messages to be read from the Shared Message Queues
- Recommendations
  - Use the OM input user exit to prevent use of NAME(*)
  - Limit use of wildcards and MSGAGE information

18

The QUERY commands do not have any impact on mainline performance, but be aware that the inclusion of wildcard characters could cause significant access to the Resource Structure or message queues. It is advisable to keep this in mind when QUERY is issued with NAME(*), which is the default.

Also, when issuing the QUERY LTERM command with the MSGAGE() parameter specified, note that all of the messages on the shared queue may be read (in determining each message's age) which could impact performance.

To assuage these possible impacts to performance, an OM input user exit can be used to prevent a QUERY command from being issued with the NAME(*) parameter specified. In addition, you can limit inclusion of wildcards and MSGAGE() information when issuing QUERY commands.

Migration/Setup Considerations

- The new type-2 commands are available automatically when Operations Manager is used
  - No option activates the new type-2 commands
- Existing type-1 and new type-2 commands can coexist in IMS 11, and in an IMSplex
  - No changes are made to type-1 command functionality
- New type-2 commands are not routed to systems in an IMSplex that are not at the IMS 11 release level
  - Only IMS 11 systems process the new type-2 commands

To use the new Type-2 QUERY TM commands, there is no special setup that needs to be done other than having Operations Manager setup and in place. In IMS 11, both Type-1 and Type-2 commands are able to be issued and the addition to the new commands in IMS 11 have no impact on the functionality of existing Type-1 commands.

Note that any system not at the IMS 11 release level will reject these new commands if they are routed to them. Only IMS 11 systems are capable of processing QUERY commands for TM resources.

Migration/Setup Considerations

- Required RACF 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

20

This table indicates how to define the RACF definitions for OM to be able to issue the new type-2 QUERY commands.

Benefits

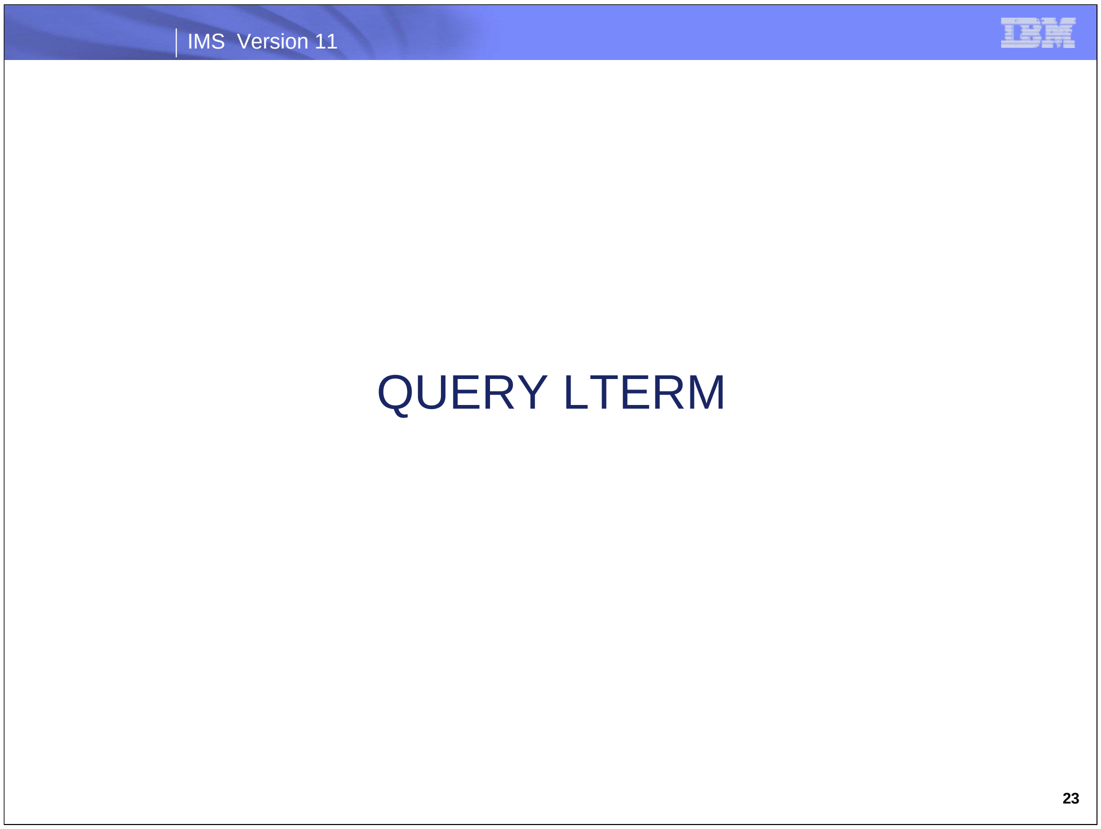
- Improve ease-of-use for managing IMS resources, by enhancing type-2 command architecture to include IMS TM resources
- Increase efficiency by consolidating output of several type-1 commands into a single type-2 command

IMS TM resources are easier to manage when the new QUERY command is used to display requested data associated with them. In addition, the output that was previously seen across multiple Type-1 commands is now included in the output of just one Type-2 command.

TM Type-2 QUERY Commands

Reference Section

The following slides are for your reference.



The following slides are for your reference.

QUERY LTERM – Equivalent 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*)

24

QUERY LTERM - Example

TSO SPOC input:

```
QRY LTERM NAME(LTERM0*)
```

TSO SPOC output:

```
Lterm    MbrName CC  Gbl
LTERM01  IMS1     0
LTERM02  IMS1     0  Y
LTERM02  IMS1     0
LTERM02  IMS2     0
LTERM03  IMS2     0
LTERM04  IMS1     0  Y
LTERM04  IMS1     0
LTERM04  IMS2     0
LTERM05  IMS1     0  Y
LTERM05  IMS1     0
```

25

In the command output you can see the “Gbl” column listed. The ‘Y’ indicates that the global status for the resource and a global status means that shared queues is being utilized. The fields in this column that are blank indicate a local status for the resource.

So as you can see, there are two IMS systems in the IMSplex: IMS1 and IMS2. Since we know that shared queues are enabled due to the “Y” indicator in the “Gbl” column, we also kno wtha the Resource Manager (RM) is maintaining status in the resource structure (STM=YES is specified in the DFSDCxxx PROCLIB member).

IMS1, the command master, displays local and global information.

IMS2 displays local information only. It ignores the second command because SHOW(GLOBAL) was specified.

LTERM01 exists on IMS1 only.

LTERM02 exists on IMS1, IMS2, and in the resource structure, is currently active on IMS2, and has no messages in the shared queues.

LTERM03 exists on IMS2 only, does not exist in the resource structure, but has one message in the shared queues.

LTERM04 exists on IMS1, IMS2, and in the resource structure, is not currently active, but is owned by IMS2 which indicates status exists on IMS2, and has three messages in the shared queues.

LTERM05 exists on IMS1 and in the resource structure, is stopped, is not currently active or owned, and has no messages in the shared queues.

LTERM06 does not exist anywhere, but has two messages in the shared queues.

QUERY LTERM - Example

TSO SPOC input:

```
QRY LTERM NAME(LTERM0*) SHOW(GLOBAL,STATUS,QCNT,OWNER)
```

TSO SPOC output:

```
Lterm     MbrName CC  Gbl   QCnt Owner  Status
LTERM02   IMS1     0  Y        0 IMS2   STATIC,RM,RMACTIVE,RMOWNED
LTERM03   IMS1     0  Y        1
LTERM04   IMS1     0  Y        3 IMS2   STATIC,RM,RMOWNED
LTERM05   IMS1     0  Y        0        STOQ,STOSEND,RM
LTERM06   IMS1     0  Y        2
```

26

There are two IMS systems in the IMSplex: IMS1 and IMS2.

The Resource Manager (RM) is maintaining status in the resource structure (STM=YES), and Shared queues are enabled.

IMS1, the command master, displays local and global information.

IMS2 displays local information only. It ignores the second command because SHOW(GLOBAL) was specified.

LTERM01 exists on IMS1 only.

LTERM02 exists on IMS1, IMS2, and in the resource structure, is currently active on IMS2, and has no messages in the shared queues.

LTERM03 exists on IMS2 only, does not exist in the resource structure, but has one message in the shared queues.

LTERM04 exists on IMS1, IMS2, and in the resource structure, is not currently active, but is owned by IMS2 which indicates status exists on IMS2, and has three messages in the shared queues.

LTERM05 exists on IMS1 and in the resource structure, is stopped, is not currently active or owned, and has no messages in the shared queues.

LTERM06 does not exist anywhere, but has two messages in the shared queues.

QUERY NODE

27

The following slides are for your reference.

QUERY NODE – Equivalent 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)

28

QUERY NODE – Equivalent Commands

Action	Type-1 Command	Type-2 keywords
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*)

QUERY NODE - Example

TSO SPOC input:

```
QRY NODE NAME(NODE2*)
```

TSO SPOC output:

```
Node     ISCUser  MbrName CC  Gbl
NODE21            IMS1     0  Y
NODE21            IMS1     0
NODE22            IMS1     0
NODE22            IMS2     0
NODE23            IMS1     0  Y
NODE23            IMS2     0
NODE24   USER24A  IMS1     0  Y
NODE24   USER24B  IMS1     0  Y
NODE24   N/A      IMS1     0  Y
NODE24   USER24A  IMS2     0
NODE24   USER24B  IMS2     0
NODE24   N/A      IMS2     0
```

30

There are two IMS systems in the IMSplex: IMS1 and IMS2.

The Resource Manager (RM) is maintaining status in the resource structure (STM=YES), and Shared queues are enabled.

IMS1, the command master, displays local and global information.

IMS2 displays local information only.

NODE21 exists on IMS1 and in the resource structure.

NODE22 exists on IMS1 and IMS2 only.

NODE23 exists on IMS2 and in the resource structure.

NODE24 is an ISC node with 3 parallel sessions available on IMS2, two of which are allocated on IMS2 and in the resource structure. IMS1 also displays an output line for NODE24 with N/A as the ISC user, which represents any global status that the node might have (that is not tied to any particular parallel session).

QUERY NODE - Example

TSO SPOC input:

```
QRY NODE NAME(NODE21) SHOW(GLOBAL,CONV,LTERM,STATUS)
```

TSO SPOC output:

```
Node     MbrName CC  Gbl  Lterm    ConvID ConvTran ConvStat Status
NODE21   IMS1     0  Y                                     CONVACT,STATIC,RM,RMACTIVE,RMOWNED
NODE21   IMS1     0  Y    LTERM21A
NODE21   IMS1     0  Y    LTERM21B
NODE21   IMS1     0  Y                   1 TRAN1A   CONVHELD
NODE21   IMS1     0  Y                   2 TRAN1A   CONVHELD
NODE21   IMS1     0  Y                   3 TRAN1A   CONVACTV
```

31

There are two IMS systems in the IMSplex: IMS1 and IMS2.

The Resource Manager (RM) is maintaining status in the resource structure (STM=YES), and Shared queues are enabled.

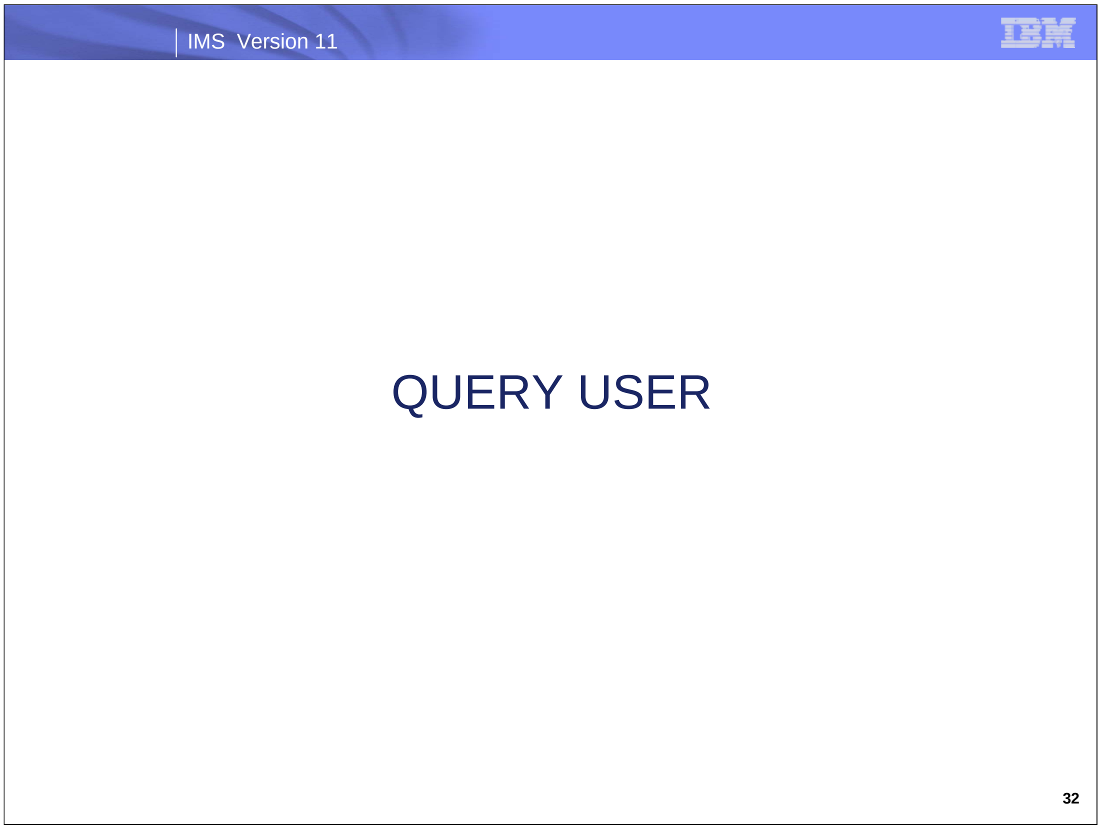
IMS1 is the command master, and because SHOW(GLOBAL) was specified, IMS1 is the only system that processes the command.

Any other IMS ignores the command (RC=4, RSN=x1000).

NODE21 exists in the resource structure. IMS1 displays a global line which shows that the node is active in the IMSplex (RM, RMACTIVE, and RMOWNED status), and has a conversation active. There are two lterms assigned to the node, and are displayed on separate output lines. There are three conversations associated with the node, and are displayed on separate output lines.

IMS1 displays an additional global line for each assigned LTERM, so 2 lines are displayed for the 2 lterms LTERM21A and LTERM21B.

IMS1 displays an additional global line for each active conversation for the node. 3 lines are displayed, because the node has 2 held conversations and 1 active conversation.



The following slides are for your reference.

QUERY USER – Equivalent Commands

Action	Type-1 Command	Type-2 keywords
Display autologon parameters	/DISPLAY USER *user* AUTOLOGON	SHOW(AUTOLOGON)
Display IMS conversation information for a particular user	/DISPLAY CONV USER *user*	SHOW(CONV)
Display message queue count in the Expedited Message Handler (EMH) queues	/DISPLAY USER *user* QCNT EMHQ	SHOW(EMHQ)
Display ISC node other half-session qualifier ID	/DISPLAY ASMT USER *user*	SHOW(ID)
Display assigned lterms	/DISPLAY ASMT USER *user*	SHOW(LTERM)
Display node	/DISPLAY ASMT USER *user* /DISPLAY USER *user*	SHOW(NODE)
Display owner IMSID in RM resource structure	/DISPLAY USER *user* RECOVERY	SHOW(OWNER)
Display preset destination	/DISPLAY USER *user*	SHOW(PRESET)
Display queue count	/DISPLAY USER *user* /DISPLAY USER *user* QCNT	SHOW(QCNT)
Display status recovery information	/DISPLAY USER *user* RECOVERY	SHOW(RECOVERY)
Display status for a particular user	/DISPLAY USER *user*	SHOW(STATUS)
Display userid	/DISPLAY ASMT USER *user* /DISPLAY USER *user*	SHOW(USERID)

33

QUERY USER – Equivalent Commands

Action	Type-1 Command	Type-2 keywords
Display user resource version number assigned by the Resource Manager	N/A	SHOW(VERSION)
Display user with active or held conversations	/DISPLAY CONV	STATUS(CONV)
Display user with specified status	/DISPLAY STATUS USER	STATUS(*status*)

QUERY USER - Example

TSO SPOC input:

```
QRY USER NAME(USER25) SHOW(GLOBAL,CONV,LTERM,STATUS)
```

TSO SPOC output:

```
(screen 1)
User     MbrName CC  Gbl  Lterm    ConvID ConvTran ConvStat
USER25   IMS1     0  Y
USER25   IMS1     0  Y    LTERM25A
USER25   IMS1     0  Y    LTERM25B
USER25   IMS1     0  Y                  1 TRAN1A   CONVHELD
USER25   IMS1     0  Y                  2 TRAN1A   CONVHELD
USER25   IMS1     0  Y                  3 TRAN1A   CONVACTV
(scrolled right to screen 2)
User     MbrName Gbl  Status
USER25   IMS1    Y    CONVACT,RM
USER25   IMS1    Y
USER25   IMS1    Y
USER25   IMS1    Y
USER25   IMS1    Y
USER25   IMS1    Y
```

35

RM is maintaining global status in the resource structure (STM=YES).

IMS1 is the command master, and because SHOW(GLOBAL) was specified, IMS1 is the only system that processes the command.

Any other IMS ignores the command (RC=4, RSN=x1000).

USER25 exists in the resource structure.

The first line displays the global status of USER25, which indicates it exists in the resource structure (RM status), and has a conversation active (CONVACT status). The user is not currently signed on or owned by any IMS because there is no RMACTIVE or RMOWNED status.

IMS1 displays an additional global line for each assigned LTERM, so 2 lines are displayed for the 2 lterms LTERM25A and LTERM25B.

IMS1 displays an additional global line for each active conversation for the user. 3 lines are displayed, because the user has 2 held conversations and 1 active conversation.

QUERY USERID

The following slides are for your reference.

QUERY USERID – Equivalent 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)

37

QUERY USERID - Example

```
TSO SPOC input:
QRY USERID NAME(USERID*) SHOW(ALL)

TSO SPOC output:
(screen 1)
UserID   MbrName CC Gbl Owner  Node     User     Version#
USERID01 IMS1     0 Y   IMS1   NODE01   USER01          1
USERID01 IMS1     0
USERID02 IMS1     0 Y   IMS2   NODE02   USER02          1
USERID02 IMS2     0
USERID03 IMS1     0 Y   IMS1   NODE03                   1
USERID03 IMS1     0

(scrolled right to screen 2)
UserID   MbrName Gbl  Status                LNode    LUser
USERID01 IMS1     Y    RM,RMACTIVE,RMOWNED
USERID01 IMS1                               NODE01   USER01
USERID02 IMS1     Y    RM,RMACTIVE,RMOWNED
USERID02 IMS2                               NODE02   USER02
USERID03 IMS1     Y    RM,RMACTIVE,RMOWNED
USERID03 IMS1                               NODE03
```

38

There are two IMS systems in the IMSplex: IMS1 and IMS2.
RM is maintaining global status in the resource structure (STM=YES).
Single-signon for user IDs is enforced (otherwise, userids would not exist in RM).
IMS1, the command master, displays global and local information.
IMS2 displays local information only.

USERID01 is signed onto dynamic or ISC node NODE01, user USER01 on IMS1.
USERID02 is signed onto dynamic or ISC node NODE02, user USER02 on IMS2.
USERID03 is signed onto static node NODE03 on IMS1.

RM status indicates the userid exists in the RM structure.
RMACTIVE indicates the userid is currently signed on in the IMSplex.
RMOWNED indicates the userid is owned by an IMS in the IMSplex.

DFSMSCE0 Enhancements

DFSMSCE0 Enhancements

- Support for IMSplex affinity routing
- Addition of a new exit entry point
  - During application GU

IMSplex Affinity Routing

IMSplex Affinity Routing

- Extension of the routing capabilities of the TM and MSC Message Routing and Control User Exit routine (DFSMSCE0)
  - To support affinity routing in an IMSplex environment
    - Ability to establish an affinity between a transaction message and an IMSID
      - Where affinity processing selects a single IMS to process the transaction.
- Addresses difficulties when migrating to an IMSplex environment
  - Routing a message to the correct IMS backend system in ways other than the use of region and transaction class scheduling
  - Overriding APPC/OTMA affinity restrictions
  - Implementing familiar routing capabilities for those moving from MSC environments

42

The IMSplex Affinity Routing support extends the capabilities of the user exit DFSMSCE0 to control and establish affinities when routing messages to back end (BE) IMS's within a shared queues (SQ) IMSplex. This is accomplished by allowing the exit to specify an IMSID that can be appended to the destination SQNAME thereby establishing an affinity between the transaction message to the IMSID.

IMS Version 11

Implementation

- Initial Support – for tooling
  - Enhancement to support affinity between a transaction message and a specific IMS in an IMSplex environment
    - DFSMSCE0 entry points (Terminal, Program and Link Receive)
      - Support affinity routing by appending the 4 character IMSID to the message shared queues name (DFSSQNM macro)
      - IMS Commands to register the transaction for affinity processing
      - PK55461/UK36861/UK36862 (IMS 9), PK55462/UK36864/UK36865 (IMS 10)

- IMS 11
  - Extends the initial support and provides additional enhancements:
    - Support for a 7 char XRF RSENAME for IMSID
    - APPC/OTMA forced affinity enhancement
    - Expansion of the DFSMSCE0 input parameter list (DFSMSCEP)
    - New command keyword - AFFIN

43

The initial implementation of the IMSPLEX Affinity Routing support was provided as a set of small programming enhancements to IMS 9 and IMS 10 primarily in support of the Sysplex Manager Tool and other vendor tools. By allowing the three routing entry points (Terminal Routing, Program Routing, and Link Receive Routing) of the DFSMSCE0 user exit routine to support appending an IMSID to transaction messages, a shared queues IMSplex environment can support an affinity between the transaction message and an IMS system associated with the selected IMSID. Note that the affected transactions must also be registered for affinity processing in the target IMS system. For non shared queues systems, this affinity is ignored.

IMS 11 extends the initial support by adding the following:

- Expansion of the MSCEAFIN field to 8 bytes to support XRF environment
- Support for APPC/OTMA environments
- Expansion of DFSMSCEP input parameter list since it is out of space.
- Addition of the status AFFIN to the show transaction status command to show transactions that are registered for affinity

DFSMSCE0 Exit Routine Enhancements

- DFSMSCE0 exit input parameter list (mapped by DFSMSCEP macro)
  - 3 flags documented as, “Exit requests message be processed in a BE IMS (this option currently not implemented)” are now able to process requests
    - MSTRFL2 = MSTR2BSQ - for terminal routing
    - MSLRFL2 = MSLR2BSQ - for link receive routing
    - MSPRFL2 = MSPR2BSQ - for program routing
  - Affinity field
    - MSCEAFIN - 8 characters (IMS 11) to support 7 character XRF RSENAME
      - Initial support (IMS 9/ IMS 10) only allowed appending 4 character IMSID

44

To invoke affinity routing the user exit routine must modify the DFSMSCE0 exit input parameter list, mapped by macro/DSECT DFSMSCEP, to set one of three flags as follows: MSTR2BSQ for terminal routing, MSPR2BSQ for program routing, and MSLR2BSQ for link receive routing. Additionally the routine must copy the IMSID/XRF RSENAME to the affinity field MSCEAFIN (padded with blanks if needed). This field, in the initial implementation, only supported the 4 character IMSID but in IMS 11 has been expanded to 8 characters to accommodate the 7 character XRF RSENAME. When the user exit returns to IMS, the request is validated and if OK, the value in the MSCEAFIN is appended to the messages shared queues name (i.e. mapped by DFSSQNM macro/DSECT).

DFSMSCE0 Exit Routine Enhancements …

- DFSMSCE0 exit input parameter list …
  - MSCEFL2 = MSCE2FAF (IMS 11)
    - Primarily needed for synchronous APPC/OTMA transactions
      - Forces affinity when either the APPC/OTMA shared queues enablement option (AOS=N) or resource recovery support (RRS=N) are not active
        - AOS=N/RRS=N ordinarily prevent synchronous processing in the back-end
    - IMS maintains the connection to the APPC/OTMA client on the front-end system while processing the message as a non-APPC/OTMA message within the IMSplex environment
      - MSC must be defined
  - MSPRFL4 = MSPR4OUT
    - New program routing flag that indicates the application program calling the exit is a non-message driven BMP with the OUT= ltermname/trancode
      - These messages are inserted to the I/O PCB
        - DFSMSCE0 previously did not support I/O PCB routing

45

The next enhancement supports a new flag, MSCEFL2 = MSCE2FAF, to force affinity for APPC synchronous or OTMA Commit Mode 1 transactions where IMS restricts the affinity routing request because either AOS=N or RRS is not used, RRS=N. When the flag is on, IMS disconnects the message from its APPC/OTMA client conversation and processes it asynchronously in non-APPC/OTMA mode until the response is received. At that time, the response message is converted back to APPC/OTMA synchronous mode, and the response is queued to the client. Note: The ability to process an APPC/OTMA transaction in non-APPC/OTMA mode requires the MSC feature to be defined (sysgened) within the IMSplex to allow storage of the APPC/OTMA information in the message's MSC extension prefix while the conversation is disconnected.

Additionally, if the affinity request is valid (APPC ASYNC, OTMA CM0, or APPC SYNC/OTMA CM1 and AOS=Y| F, RRS=Y) or the transaction is not APPC or OTMA, IMS ignores this flag and processes the message in normal synchronous conversation mode. The affinity continues to be honored.

The new MSPR4OUT flag indicates that the calling program is a BMP with the OUT parameter specified. The enhancement supports I/O PCB routing.

DFSMSCE0 Exit Routine Enhancements …

- DFSMSCE0 exit parameter list (DFSMSCEP)
  - Has more documentation on the new fields
    - Assemble and refer to flags MSTR2BSQ, MSLR2BSQ, MSPR2BSQ, MSCEAFIN, MSCE2FAF, MSPR4OUT
  - Has been expanded
    - 20 bytes reserved (available) space for each exit interface (terminal, link receive, and program routing)
  - Has a new version number
    - Field MSCEPLVER is 004 (in IMS 10 the field was 003)
- DFSMSCE0 sample exit
  - Has examples of back end (BE) affinity routing
  - Assemble and refer to flags MSTR2BSQ, MSLR2BSQ, and MSPR2BSQ

46

The DFSMSCEP parameter list has been expanded with reserved fields for future enhancements. Each routing entry point (i.e. the main entry, and the terminal, link receive, and program routing entries, and the initialization, and termination entry points) includes 20 bytes of reserved space. Note that field MSCEPLVER has been updated to 0004 from the IMS 10 value of 0003.

Information on the exit possible actions and examples of back end affinity routing have been added to the DFSMSCE0 sample exit that is shipped with the IMS source library.

Affinity Registration

- Exploitation of affinity routing requires transaction registration
  - Invoke registration commands on the target IMS
    - To register a trancode for affinity processing

```
UPDATE TRAN NAME(xxxx) START(SCHD) OPTION(AFFIN)
/STA  TRANSACTION trancode AFFINITY

Example: /STA tran APOL11 AFFINITY
```

- Affinity is a status which is recovered across a warm start but not a cold start

    - Without AFFINITY, transactions are registered for normal informs

```
/START TRAN trancode
UPDATE TRAN NAME(trancode) START(SCHD)

Example:  UPDATE NAME (APOL12) START (SCHD)
```

47

Additionally, the transaction must be registered for affinity processing by issuing either: the /START TRANSACTION trancode AFFINITY command or the type 2 command, UPDATE TRAN NAME(xxxx) START(SCHD) OPTION(AFFIN) on the IMS with the selected IMSID. The registration creates the SQ queues that support the affinities. Transactions that have no affinity are registered with normal informs (i.e., normal registration with no affinity) using the same commands but without the AFFINITY option.

Note:

- Normal registration is recovered across warm starts. After a cold start, normal registration is assumed.
- Affinity registration is recovered across warm starts. After a cold start, transactions always come up without affinity registration (normal registration). A command (or tool support) will need to re-register the transaction for affinity.

If DFSMSCE0 tries to establish an affinity to a back-end IMS but the transaction is not registered for affinity, the message will be put on the shared queues with the affinity requested. In this case, with no affinity registration on the target IMS system, the message remains unprocessed and on the shared queues until a command is entered to register the transaction for affinity.

Likewise, if the transaction is started on only one IMS system and specifically started with affinity registration but DFSMSCE0 routes the message without affinity, the message also remains unprocessed until the transaction is started on an IMS system without affinity, i.e., a normal inform. This shows that a transaction can be registered for both affinity and normal informs across different IMS systems in an IMSplex.

On the other hand, a transaction can be registered for affinity processing on all the IMS systems in an IMSplex. DFSMSCE0 would then have to always define an affinity for that specific transaction. The value of this scenario would be the implementation of workload balancing where DFSMSCE0 would provide the logic to round-robin the message destination with an affinity across the IMSplex.

IMS Version 11

Affinity Registration …

- Exploitation of affinity routing requires transaction registration …
  - Invoke registration commands on the target IMS …
    - To stop affinity registration

      UPDATE TRAN NAME(*xxxx*) STOP(SCHD) OPTION(AFFIN)

      - Must be issued on the IMS where the transaction was started for local affinity

    - To determine if there are messages on the SQ with affinity

      QUERY TRAN NAME(ALL) SHOW(AFFIN)
      /DIS TRAN *trancode* ALL QCNT (note: keyword ALL must be used)

      Example: /DIS TRAN ALL QCNT

```
DFS000I   TRAN      GBLQCT        AFFINITY       SYS3
DFS000I   APOL11     1               SYS3        SYS3
DFS000I       *03224* /155809        SYS3
```

48

To unregister an existing affinity processing, the UPDATE command can be issued with a STOP(SCHD) OPTION(AFFIN) request.

To determine if there are messages on the shared queues with affinity, either the QUERY or /DIS TRAN commands can be used with the ALL keyword.

Affinity Status

- AFFIN (IMS 11) - new type-2 QUERY status keyword
  - Shows if a transaction is registered for local affinity
    - Assists in determining whether or not to issue the registration command
      - QUERY TRAN(*tranname* | *ALL*) STATUS(AFFIN)
        - Shows transactions with affinity registration
      - QUERY TRAN NAME(*tranname*) SHOW(STATUS)
        - AFFIN is displayed if registered for affinity

Example: QRY TRAN NAME(APOL11,APOL12) SHOW(STATUS)

Trancode	MbrName	CC	LclStat	
APOL11	IMS1	0	AFFIN	← Registered for affinity processing
APOL12	IMS1	0		← Normal processing (non-affinity)

49

To assist the command operator in determining whether or not a transaction has already been registered for affinity processing, a new status is added to the display results of a type-2 QUERY command. AFFIN is displayed if the trancode is registered with local affinity.

This status is displayed locally only on the IMS where the affinity registration was performed, i.e. on the IMS where the /TRAN tranname AFFINITY or UPDATE TRAN NAME(tranname) START(SCHD) OPTION(AFFIN) was issued.

The example on this visual shows that APOL11 is started for affinity processing while APOL12 is started for normal processing.

Migration Considerations

- If initial support (IMS 9/ IMS 10) is already implemented
  - Ensure MSCEAFIN field is the correct length (now 8 characters) padded with blanks
- To support the expansion of the DFSMSCEP input parameter list
  - Reassemble the DFSMSCE0 exit and insure the logic is not impacted by this change
    - This affects any existing DFSMSCE0 user, even if affinity is not requested
- Command changes
  - Understand affinity registration requirements
  - Determine the impact of affinity parameters on commands (UPDATE, QUERY, /STA, /DIS) and their output (e.g., AFFIN)
- Shared queues groups with IMS 9/ IMS 10 /IMS 11
  - Requires IMS APARs on previous releases
  - No support for XRF until all IMS systems are IMS 11

50

Migration considerations include the following:

- If the IMS 9/IMS 10 initial implementation of the affinity routing support was already implemented in an IMS environment, ensure that the MSCEAFIN (formerly a 4-character IMSID value) field adheres to the new 8 character length.
- If the DFSMSCE0 exit already exists in an IMS environment, migration to IMS 11 will require that the exit be reassembled to support the expansion of the DFSMSCEP input parameter list.
- MTOs and systems personnel who issue IMS commands should be aware that affinity parameters in support of this enhancement have been added to the UPDATE, QUERY, /STA and /DIS commands and their displays.
- Shared queues groups that have mixed IMS systems will require the initial enabling APAR PK55461/ PTFs UK36861/UK36862 for IMS 9 and APAR PK55462/ PTFs UK36864/UK36865 for IMS 10. No XRF support is available until all the systems in the group are at IMS 11 level.

IMSplex Affinity Routing Support - Benefits

- Benefits
  - Additional mechanism to control transaction processing in a shared queues, IMSplex environment
    - Can be used by tools
  - Greater scheduling efficiency by eliminating the need for multiple regions to bid for a transaction message that only one region can process

Several benefits are achieved by this set of enhancements:

When DFSMSCE0 is customized to assign affinity to the transaction messages for the IMS's where they are to be processed, and the transactions are started for affinity processing, IMS notifies and process them in the selected IMS. This capability provides an additional mechanism for controlling transaction processing, over and above region and class scheduling.

Scheduling efficiency is also enhanced by eliminating the need for multiple regions to bid for a transaction message that only one region can process. Affinity processing selects a single IMS to process the transaction.

New Exit Entry Point

Additional Program Routing Entry Point

- Exit gains control on a successful application GU, IOPCB call
  - Has access to allow modification of the user prefix
    - Created if it does not already exist
    - Allows information to be added, e.g., accounting data
  - Does not support message routing (still done in CHNG, ISRT entry points)
  - IMS 10 APAR PK73423
- Implementation
  - Exit specifies PRGU parameter on the DFSMSCVT macro
    - When called, DFSMSCEP input parameter list has the following set
      - MSCEFL4=MSCE4GU indicating GU call
      - MSPRFL1=MSPR1IO indicating an IOPCB call
      - Input message address at MSCESEG
      - Input prefix address (if present) at MSCEUPR

53

A new program routing entry point is added to the DFSMSCE0 user exit which gives the exit control when an application program issues a GU to the IOPCB for the input message. The exit can modify an existing user prefix, or create a new one if it does not already exist, and request information to be added to the prefix. The intent of this capability is to provide a way for installations to include additional accounting data.

DFSMSCE0 receives control during GU processing if the PRGU parameter is specified on the DFSMSCVT macro (entry point vector table macro) along with addition of the GU entry point label of:

PROGRAM_ROUTING_GU_CALL DS 0H

Additionally, when called, the DFSMSCEP input parameter list contains indicators that the exit is being invoked for a GU call against the IOPCB. Both the input message address and input prefix address are passed to the routine.

Additional Program Routing Entry Point …

- Reminder
  - Program routing entry points
    - CHNG and ISRT – can create/modify user prefix and route messages
    - GU – can only create/modify user prefix
- Benefits
  - Provides an additional entry point during application program processing to provide installation-specific information in an architected user prefix
    - Logged as part of the message
    - Can be used by tools
  - Sample DFSMSCE0 exit shows an example of adding information

54

Again as a reminder, the GU program routing entry point cannot be used to route a message but is available to provide information in the user prefix which can be analyzed by programs that read the log.

The sample DFSMSCE0 exit routine has been enhanced to show how to add three bits of information to the user prefix: a G character indicating that the application performed a GU, the timestamp of the GU, and the transaction name. This can be modified as needed based on installation-specific requirements.

Full Function Response Mode Recovery

Full Function Response Mode Recovery *(SPE for IMS 9 and 10)*

IMS 10 APAR: PK53989; IMS 9 APAR: PK53423

- Full function response mode recovery
  - Response mode recovered across log off/log on or sign off/sign on
    - Not recovered across IMS restarts
  - Optional
    - Specified in DFSDCxxx member
- Benefit
  - User is in response mode after log on or sign on
    - Will not receive asynchronous messages before response mode output
    - Cannot enter new transactions until response mode output is received
      - These conditions can be confusing to terminal users

56

Full function response mode recovery is an option added to IMS by SPEs for IMS 9 and IMS 10. It is included with IMS 11. The SPE for IMS 10 is APAR PK53989 (PTF UK32360). The SPE for IMS 9 is APAR PK53423 (PTF UK32266).

This option recovers full function response mode across a log off and log on for a static terminal or a sign off and sign on for a dynamic terminal user. It is specified in the DFSDCxxx member. Details are on the following page.

Without this option, users may be confused by the messages that they receive after they lose their connection to IMS and then log on or sign on. First, the response to their previous input is queued as an asynchronous message. This means that it is delivered after any other messages on the queue. Second, if they enter another response mode transaction before they receive the queued output, the response for the transaction entered before the loss of connection will be returned some time later. It may not be obvious to the terminal user why it is delivered then. The full function response mode recovery option eliminates these confusing conditions. When the option is used, the message returned after the log on or sign on is the response to the previous input.

Full Function Response Mode Recovery

- Specifying full function response mode recovery
  - Option specified in DFSDCxxx PROCLIB member
    - RCVYRESP=YES|NO
    - RCVYRESP=YES applies only to users and terminals with LOCAL status recovery mode
      - SRMDEF=LOCAL must be specified or assigned by a descriptor or exit routine
  - Cannot be used with global status recovery mode (SRMDEF=GLOBAL)
  - Cannot be used with sysplex terminal management (STM=YES)

57

The full function response mode recovery option is chosen by specifying RCVYRESP=YES in the DFSDCxxx PROCLIB member. RCVYRESP=NO is the default. RCVYRESP=YES is invalid when SRMDEF=GLOBAL is specified.

When SRMDEF=NONE is specified in the DFSDCxxx PROCLIB member, some users and terminals may be assigned SRM=LOCAL through a user descriptor or the DFSSGNX0 or DFSLGNX0 exit routines. In these cases, if RCVYRESP=YES is also specified, full function response mode recovery will apply to these users and terminals. When SRMDEF=NONE is specified, users and terminals not assigned SRM=LOCAL by a descriptor or exit routine will not have full function response mode recovery. In other words, RCVYRESP=YES applies to all terminals and users for which LOCAL status recovery mode applies. It does not apply to any terminal or user for which GLOBAL status recovery mode or no status recovery mode applies.

Shared Queues

False Scheduling Reduction

Shared Queues False Scheduling Reduction

- False schedules are reduced
  - False schedules caused by a transaction reaching the PARLIM count will be reduced
    - IMS will not schedule another region if another message has not arrived
  - Different installations will see different improvements
- Benefits
  - Reduced resource consumption

59

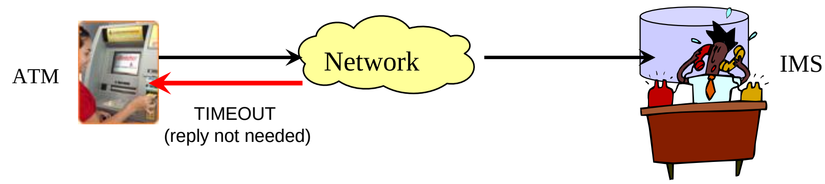
False schedules are schedules of message regions for which no message is available when they first do a GU call. This can occur in two ways. First, another IMS system in the shared queues group may have retrieved the message. Second, after each successful GU for the transaction IMS determines the current number of consecutive successful GU calls for the transaction. If this number is greater than the product of the PARLIM value and the number of regions currently scheduled for this transaction in this IMS system, IMS attempts to schedule the transaction in another region. If there are no more messages, the schedule will be a false schedule. Since many shared queues installations use PARLIM values of 0 or 1, this may occur frequently. IMS 11 has changed the processing for the second case. It will not schedule the transaction in another region if another message has not arrived.

This enhancement reduces the probability of these false schedules. The benefit will vary by installation.

Transaction Expiration

Transaction Expiration (Input Message Timeout)

- New EXPRTIME attribute for transactions
  - Allows input messages to expire and be deleted prior to processing
- Addresses the situation when unexpected delays in the network or in IMS result in a remote client timeout before receiving a reply message
  - Response is no longer required but the transactions remain queued in IMS for processing



If a message takes a long time to process in IMS, the client could time out or a network failure could occur before the message is processed. In such conditions, when IMS schedules the transaction and retrieves and processes the message, the output response may no longer be wanted or needed. IMS 11 addresses this situation with a new EXPRTIME attribute for transactions that provides a mechanism to optionally request the expiration of transaction input messages and allow IMS to discard messages prior to processing.

Specification Levels

- Transaction level specification
  - System definition – TRANSACT macro EXPRTIME
  - Output Creation Exit – DFSINSX0
  - DRD - CREATE TRAN/TRANDESC, and UPDATE TRAN/TRANDESC
    - Resource Definition Data Set (RDDS) support

- Message level specification
  - Applicable only to messages that are sent to IMS through OTMA
    - Explained in greater detail in the OTMA section
  - Expiration override and extended actions on an individual message level

IMS provides two levels of message expiration specification.

The transaction level which applies to all IMS messages allows the expiration value to be defined through system generation in a new attribute in TRANSACT macro, or through the IMS DRD type-2 commands: CREATE TRANS|TRANDESC and UPDATE TRANS|TRANDESC, or through the Output Creation Exit Routine DFSINSX0.

The message level specification which only applies to the OTMA environment allows the message expiration time to be specified in the OTMA message prefix. This level of support is documented in the OTMA Enhancements section of the material and will not be covered here.

IMS Version 11

Transaction Level Expiration

Information in this section Only refers to Transaction Level Specification of expiration time

- Transaction level expiration time
  - A value, in seconds, that IMS uses to compare against the elapsed time of an unprocessed transaction input message
    - Values are compared when the scheduled dependent region retrieves the input message via GU
- IMS actions for expired transaction message conditions:
  - IMS 0243 Abend, and
    - DFS555I TRAN ttt ABEND …
      - Displayed when an expired message is detected, OR
    - DFS2224I TRANSACTION ON A BACK-END SYSTEM ABENDED
      - Displayed when an expired OTMA or APPC message is detected on a back-end shared queues system
    - 67D0 log record

63

Note that the remainder of the discussion in this section applies only to the Transaction Level Specification support since it applies to all IMS messages regardless of the mechanism used to send the message to IMS.

IMS compares the elapsed time of an unprocessed message against the transaction expiration value. The elapsed time used for the comparison includes the time when the message arrives in IMS (input timestamp) until the time when the message is scheduled into a dependent region and the region issues a GU to retrieve the message. If the input message is considered expired then it is deleted and the transaction is abended with a 0243 abend. Additionally, depending on whether the environment is shared queues or not and whether or not the message is from APPC or OTMA, either a DFS555I or DFS2224I message is sent to the end user.

The 0243 abend description has been enhanced in IMS version 11 for this new transaction expiration support:

0243:

An APPC transaction has been timed out by IMS TM. The time-out value is specified in the APPCIOT=(,XX) parameter, or

IMS has detected an expired input transaction and has canceled the transaction. The expiration time for the cancelled transaction could have been specified in the OTMA message prefix, via the TRANSACT macro, via the IMS destination creation exit DFSINSX0, or via DRD type-2 commands such as CREATE/UPDATE TRAN commands.

System Action: No storage dump is created and the dependent region controller is reattached.

Implementation

- System Definition – TRANSACT macro
  - **EXPRTIME**= 0 – 65535
    - Value in seconds that IMS uses to compare against the elapsed time of an unprocessed message
      - Messages that exceed the value are considered expired and will not be processed
      - 0 means no expiration time is set
    - Specification of an invalid value results in warning message
      G316 EXPRTIME OPPERAND INVALID, DEFAULT OF ZERO ASSUMED
- DFSINSX0 Output Creation Exit Enhancement
  - DSECT INSXTRNQ includes the new transaction expiration attribute
  - Can be used when dynamically creating a transaction destination
    - Attribute applies to transactions scheduled on the local system and any shared queues back-end systems

64

When using the system definition process, the TRANSACT macro provides a new EXPRTIME parameter with valid elapsed time values ranging from 0 to 65535 seconds. The default value of 0 turns off the support. IMS uses the specified value, from 1 to 65535 seconds, to compare against the elapsed time of an input message. The specification of an invalid value results in a warning message G316 being issued, a return code of 2, and an automatic reset of the value to the default of zero. During execution, input messages that exceed the specified value are considered expired and will not be processed.

The output destination creation exit DFSINSX0 can also be used to dynamically create a transaction. The new EXPRTIME transaction expiration attribute applies to transactions scheduled on both the local system as well as any shared queues back-end system. The DSECT INSXTRNQ will be updated to set the new transaction expiration attribute

IMS Version 11

Implementation …

- Dynamic Resource Definition (DRD)
  - DRD commands
    - CREATE TRAN|TRANDESC **SET (EXPRTIME(xxxx))**
      - Where EXPRTIME = 0 - 65535
    - UPDATE TRAN|TRANDESC **SET (EXPRTIME(xxxx))**
      - Updates the EXPRTIME value and overrides the TRANSACT macro and any CREATE command specified values
    - QUERY TRAN|TRANDESC **SHOW (EXPRTIME | ALL)**
  - EXPRTIME value is logged in the X’22’ log record
    - Associated with the CREATE and UPDATE commands
    - Recovered over warm and emergency restarts
    - Propagated to XRF systems

65

The DRD CREATE and UPDATE commands support the new EXPRTIME attribute and can be used to set the transaction expiration time for a transaction or a group of transactions. Note that UPDATE command specifications override the value in the TRANSACT macro and/or the value defined with the CRE command. Additionally, the QUERY command can be used to display the specified expiration time for a transaction.

This EXPRTIME attribute is logged in the x’22’ log record for CREATE and UPDATE commands and is therefore recoverable over warm/emergency restarts and also propagated to XRF alternate system.

Implementation …

- RDDS Extraction Utility enhancement for the new EXPRTIME transaction attribute

  – New attribute is exported to the RDDS and can be imported to another IMS via the IMPORT command

    - EXPORT DEFN TARGET(RDDS) TYPE(ALL|TRAN|TRANDESC) command exports the transaction expiration time to the RDDS

    - IMPORT DEFN SOURCE(RDDS) TYPE(ALL|TRAN|TRANDESC) command or IMS auto-import processing imports the transaction expiration time from the RDDS to the IMS

The RDDS Extraction utility has been expanded to recognize the EXPRTIME attribute which can be exported to an RDDS dataset and subsequently imported to another IMS via the IMPORT command. Both the EXPORT DEFN TARGET(RDDS) TYPE(ALL|TRAN|TRANDESC) and IMPORT DEFN SOURCE(RDDS) TYPE(ALL|TRAN|TRANDESC) commands as well as the IMS auto-import processing include the EXPRTIME value.

Restrictions

- The following messages are not part of the support
  - IFP, MSC, Conversational txns and switched to txns on a pgm-pgm switch

Note that a restriction applies to messages associated with IFP, MSC, Conversational transactions, and switched to transactions on a pgm-pgm switch.

Transaction Expiration - Benefits

- Benefits
  - Reduces unnecessary processing costs by preventing the execution of transactions that have aged beyond a certain time and are no longer needed
    - Frees resources to be used by other messages

The transaction expiration function reduces unnecessary processing costs and CPU cycles associated with expired transaction messages and potentially frees up resources for use by other applications.

Transaction Manager

- Type-2 Query Commands
- DFSMSCE0 Enhancements
- Full Function Response Mode Recovery
- Shared Queues False Scheduling Reduction
- Transaction Expiration


IMS Version 11
Connectivity
Information Management software
© 2008 IBM Corporation

Connectivity

- OTMA
  - Consistency enhancements for Shared Queues environments
  - Timeout enhancements
  - Resource Monitoring
  - Usability
- IMS Connect
  - IMS Connect configuration member HWSCFGx enhancements
  - Enhanced commands
  - Exit Routine enhancements
  - Cancel Client ID
  - TCP/IP Auto Reconnect
  - Generated Client ID
  - Performance enhancement
  - New Recorder Trace and BPE support

OTMA Enhancements

IMS Version 11

OTMA Enhancements

- Consistency enhancements for Shared Queues environments
  - Error message handling
  - Super member extension
- Timeout enhancements
  - Transaction expiration – input message timeout
  - Enhanced timeout for hung TPIPEs
- Resource Monitoring
  - Flood detection and early failure notification
- Usability
  - Type-2 DRD support for OTMA descriptors

4

The OTMA enhancements in IMS 11 address consistency, processing cost, resiliency and usability.

Consistency Enhancements for Shared Queues Environments

Error Message Handling

- Greater consistency in transaction abend error message support for CM0 (commit-then-send) interactions
  - DFS555I message from a back-end shared queues system is queued and sent to the front-end IMS for the OTMA client
  - PK35745/UK31054 (IMS 9), PK38720/UK31057 (IMS 10),
- Addresses situation
  - Remote client hang waiting for CM0 output until timeout
    - DFS555I messages not sent from back-end system
- Benefit
  - Provides consistency for IMS shared queues environments so that the result of the abend in either front-end IMS or back-end IMS is the same

6

Prior to this capability, OTMA asynchronous support for Shared Queues environments documented a restriction relating to a back-end system abend situation. Transaction abends on a back-end systems did not support sending the DFS555I message to the waiting remote client attached to a front-end IMS. The remote clients had to implement a timeout flow to ensure that the connection would eventually be broken.

IMS 11 along with IMS 9 (PK35745/UK31054) and IMS 10 (PK38720/UK31057) lift the restriction and allow the DFS555I message to flow from the back-end IMS through the front-end to the waiting remote client application.

Shared Queues ALTPCB back-end Support

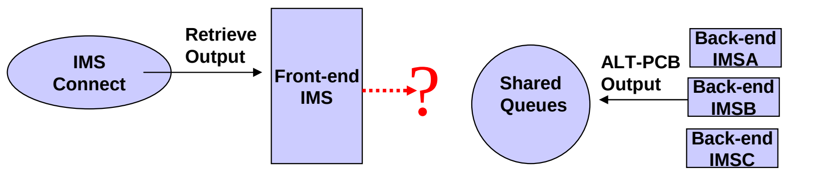
- Enhancement to OTMA super member support
  - Allows IMS Connect Resume TPIPE request to retrieve an ALTPCB message from a Shared Queues back-end system
    - Resume TPIPE AUTO and SINGLE-WAIT requests
  - PK56730/UK35483 (IMS 9), PK61174/UK44768 (IMS 10), PK80945 UK51334 (IMS 11)

- Addresses situation
  - When ALTPCB output message are generated on one of the back-end SQ systems
    - And the remote client has no knowledge from which IMS the retrieval should be requested
      - Super member support initially only allowed retrieval of messages that were already on the queue
        - Did not support creating a program that could automatically wait for new messages

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, even using super member support, to create a listening remote client that could wait for and retrieve these types of messages as they are queued was previously documented as a restriction.

Background

- Resume TPIPE requests and Shared Queues
  - Original problem without super member support
    - Resume TPIPE request must be sent to the specific IMS that produced the ALTPCB output



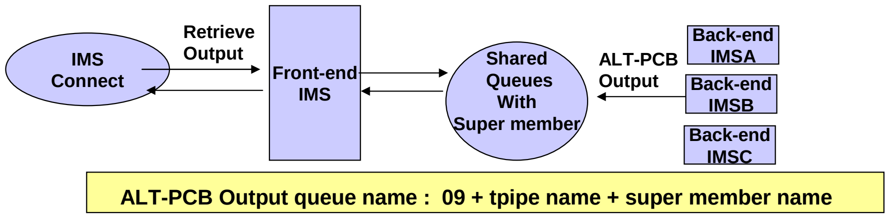
Message has an affinity to the back-end:
ALTPCB Output queue name : 09 + backend Qab Token + backend IMSid

8

The original problem for the Shared Queues environment without super member support is that when an ALTPCB message is created in a back-end IMS, it is created with a queue name that includes the imsid of the system in which it was produced. This affinity makes it difficult for a remote client to know which IMS system should be named for the Resume TPIPE request.

Background …

- Resume TPIPE requests and Shared Queues …
  - With the addition of super member support
    - Resume TPIPE request can be issued on any IMS system



- BUT with a restriction
  - Only messages that are already queued can be retrieved
    - SINGLE or NOAUTO

9

With the addition of super member support, the shared queues name was changed to use the generic super member name rather than the specific imsid. This capability allowed any IMS in the Shared Queues group to support a Resume Tpipe that retrieved a queued message from the back-end system. This support, however, came with a documented restriction that only messages already on the queue could be retrieved. In other words, Resume Tpipe with SINGLE or NOAUTO.

Background …

- But since there are two basic types of Resume TPIPE requests
  - (1)Retrieve messages already on HOLD queue
    - SINGLE or NOAUTO (returns multiple messages one at a time)
  - (2) Wait for messages to be added to the queue
    - SINGLE-WAIT or AUTO
- The remaining issue
  - Initial super member capability only supported (1)
    - An ALTPCB messages already on the queue or an IOPCB message that had previously failed delivery
  - No support for (2) by a remote server “listener” or “wait for input” type of program
    - SINGLE-WAIT converted to SINGLE and AUTO converted to NOAUTO

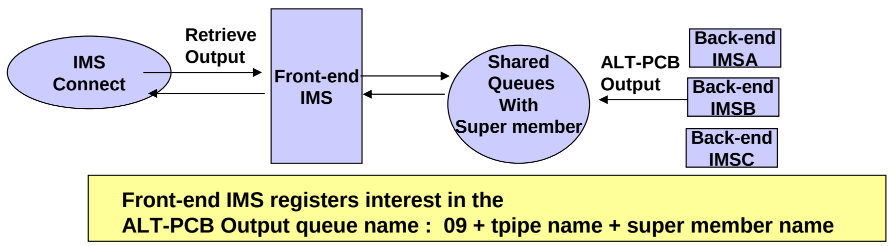
10

An issue still remained because there are two types of Resume TPIPE requests: requests for messages that are already on the hold queue and requests for messages that do not yet exist but will be put on the queue later.

Because the initial super member support was restricted to NOAUTO or SINGLE, any request that carried the flags for either AUTO or SINGLE-WAIT were converted to NOAUTO and SINGLE respectively. This meant that the Resume TPIPE could only retrieve an ALTPCB output message that was already waiting to be delivered or an IOPCB message that had previously failed delivery. This support, therefore, precluded the ability to create a “listener” or “wait for input” type of server program that could retrieve messages as they are created.

Enhanced Shared Queues ALTPCB back-end Support

- Therefore, a new capability
  - Extends the existing super member function to support Resume TPIPE AUTO and SINGLE-WAIT request
    - Registration of interest in the super member TPIPE queue at the front-end



11

The OTMA enhancement for SQ ALTPCB messages utilizes the shared queues notification method to inform the front-end IMS when ALTPCB output is generated at the shared queues back-end system. The front-end IMS registers interest in the super member TPIPE queue so that it can be notified when a message is available and is therefore able to retrieve the ALTPCB output messages from the super member queue for an OTMA client.

IMS Connect has also been enhanced as part of this support to handle a NAK associated with a Resume TPIPE request. In previous releases, IMS Connect clients that NAK'ed a Resume TPIPE message were returned to CONN state. Because OTMA did not send any additional messages after the NAK, the remote client had to wait for a timeout or be canceled (with STOPCLNT or Cancel Timer). This enhancement allows IMS Connect to determine if the NAK is associated with a Resume TPIPE request that did not specify Reroute. In this case the remote client is immediately timed out, sent an RSM RC='28'x indicating the timeout and then placed into RECV state so that additional messages can be processed.

Enhanced Shared Queues ALTPCB back-end Support …

- As well as an enhancement to IMS Connect to handle Resume TPIPE AUTO and NAK
  - IMS Connect clients are immediately timed out, sent an RSM RC=x'28', and put back into RECV state
  - Previously, remote client was returned to CONN state until either a timeout or cancel occurred
  - PK60117/UK35225 (IMS 9), PK61673/UK35955 (IMS 10)

12

IMS Connect has also been enhanced as part of this support to handle a NAK associated with a Resume TPIPE request. In previous releases, IMS Connect clients that NAK'ed a Resume TPIPE message were returned to CONN state. Because OTMA did not send any additional messages after the NAK, the remote client had to wait for a timeout or be canceled (with STOPCLNT or Cancel Timer). This enhancement allows IMS Connect to determine if the NAK is associated with a Resume TPIPE request that did not specify Reroute. In this case the remote client is immediately timed out, sent an RSM RC='28'x indicating the timeout and then placed into RECV state so that additional messages can be processed.

Enhanced Shared Queues ALTPCB back-end Support …

- Benefit
  - Allows remote program to “listen” or “wait” for a message to be queued
    - Without concern as to which front-end IMS is used for the client connection
    - Simplifies remote program logic
  - Enhances the generic routing and IP spraying support that IMS Connect provides for TCP/IP clients that access IMS environments

Transaction Expiration
(Input message timeout)

IMS Version 11

Highlights

- Transaction expiration attribute = input message timeout
  - Allows input messages to expire and be deleted prior to processing
- Transaction level specification
  - Discussed previously in the TM Enhancements section
    - System definition – TRANSACT macro EXPRTIME
    - Output Creation Exit – DFSINSX0
    - DRD - CREATE TRAN/TRANDESC, and UPDATE TRAN/TRANDESC
      - Resource Definition Data Set (RDDS) support

15

The transaction expiration support allows input messages whose time in IMS has exceeded an expiration value to be discarded prior to processing. The EXPRTIME attribute associated with the transaction provides the value that IMS uses to make the determination.

As described in the Systems Enhancements material, the transaction level which applies to all IMS messages allows the expiration value to be defined through system definition in a new attribute in the TRANSACT macro, or through either the IMS DRD type-2 commands: CREATE TRANS|TRANDESC and UPDATE TRANS|TRANDESC, or through the Output Creation Exit Routine DFSINSX0.

Highlights …

Enhanced support only for OTMA messages:

- Message level specification
  - Expiration time overrides and extended actions on an individual message level
    - Allows an OTMA message to carry an expiration value using one of two methods:
      - An expiration STCK time
        - Supported by IMS Connect
      - An elapsed time for transaction, similar to EXPRTIME value
  - Message level for OTMA messages is an IMS 10 SPE
    - PK74017/UK50901 (IMS 10), PK74024/UK50910 (IMS Connect)
  - Transaction level remains an IMS 11-only capability

16

The OTMA environment supports two levels of specifying when an unprocessed input message should expire: the transaction level specification as well as enhanced support for message level specification.

Note that message level specification for the expiration value is supported only for OTMA interactions. When specified in the OTMA prefix, the expiration value overrides any other value that may have been provided through the transaction level specification options as documented on the previous visual. OTMA supports two formats for message level expiration times: a STCK format and an elapsed time format. OTMA clients can choose to use one or the other format for individual messages. IMS Connect implements the STCK format and a future MQ capability will take advantage of the elapsed time format.

OTMA Message Level Specification

1. STCK format in the input message

   – Allows OTMA to compare the current STCK time in the processing cycle with the specified expiration STCK time that is passed into IMS

     • Implementation

       –OTMA State Data Prefix
         • TMAMTXP1 (x'01') flag in the TMAMHIST byte requests STCK format
         • TMAMOSXP specifies offset to STCK value in the user data prefix

       –OTMA User Data Prefix
         • contains STCK value to be used for the expiration decision

17

The first message level format supports the specification of the transaction expiration time in STCK format as part of the OTMA prefix in the input transaction message. When received using this capability, OTMA compares the current STCK time with the input STCK expiration time to make the decision whether or not to expire the input transaction.

To take advantage of the STCK message format, OTMA client implementations include the following settings in the OTMA prefix:

- State Data Prefix

TMAMTXP1 flag (x'01') in the TMAMHIST byte to request the STCK support

TMAMOSXP to specify the offset in the User Data Prefix that can be used to locate the STCK value

- User Data Prefix

STCK value for the expiration time that OTMA for comparison against the current STCK value

OTMA Message Level Specification…

1. STCK format in the input message (cont'd)…

– ***IMS Connect support***

- New IRM flag activates the transaction expiration function
  - –IRM_F1_TRNEXP in the IRM_F1 byte

- ***IMS Connect does the rest***
  - –Computes the transaction expiration time in STCK format based on current time and either of the existing IMS Connect timeout specifications:
    - The IRM_TIMER value that is specified in the IRM if it has been specified, or
    - The TIMEOUT value in the HWS configuration file.
  - –Stores the computed STCK time in the OTMA User Data Prefix
  - –Sets the flag and provides the offset to the STCK time in the State Data Prefix

By computing a transaction expiration value based on either the IRM_TIMER or the HWSCFGxx TIMEOUT value, IMS Connect simplifies the problem of attempting to coordinate these values with an IMS transaction expiration timeout.

OTMA Message Level Specification…

1. STCK format in the input message (cont'd)…

   – IMS Connect Consideration
     - IRM_TIMER value versus HWS Configuration TIMEOUT value
     - Overrides the OTMA transaction level specification

- Benefit
  - Assists in the coordination of timeout specifications

This coordination is important because timeouts in the processing path of a message should be set using values that do not conflict or cause a timeout in one component without consideration of the processing flow in another component. For example, if the transaction expiration timeout in IMS is much longer than the IRM_TIMER, then the IMS Connect environment may experience a timeout while a delayed message in IMS could still process when it is no longer needed. By coordinating both values, a timeout in IMS Connect will also result in a transaction expiration timeout.

OTMA Message Level Specification…

2. Elapsed time format in the input message

   – Allows the input message to provide an elapsed expiration time in seconds that can be used by OTMA to determine whether or not the message should be expired prior to processing

     - Implementation
       - –State Data Prefix
         - TMAMTXP2 (x'02') flag, in the TMAMHIST byte activates the transaction expiration function
         - TMAMOSXP 2-byte field specifies the transaction expiration time in seconds

20

The second message level format supports the specification of the transaction elapsed in the input message. IMS OTMA uses this value to determine when a message should be expired.

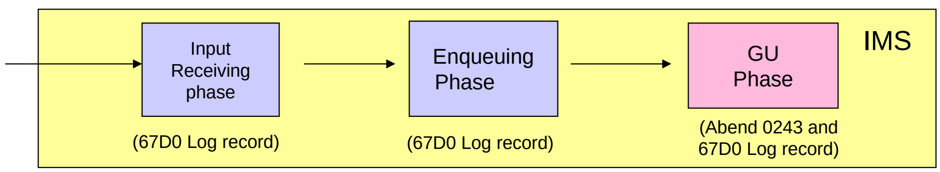
To take advantage of this second message format, OTMA client implementations include the following settings in the OTMA State Data Prefix:

- TMAMTXP2 (x'02') flag in the TMAMHIST byte
- TMAMOSXP field to specify the elapsed time that IMS OTMA can use to calculate when to expire the transaction message

IMS Version 11

OTMA Transaction Expiration Processing

- The checks to determine input message expiration occur during:
  - Input Receiving Phase (OTMA only)
    - When the input is received
  - Enqueuing Phase (OTMA only)
    - Before OTMA enqueues the transaction
  - GU Phase (OTMA and non-OTMA)
    - When the message is scheduled into a dependent region and a GU is processed
      - This phase is the only check for non-OTMA input messages



NOTE: IFP, MSC, IMS Conversational transactions and switched-to transaction for pgm-pgm switched messages are only checked for expiration during the Input Receiving and Enqueuing Phases

21

OTMA determines whether an input message has reached its expiration value and be discarded in one of three processing phases:

- Input Receiving Phase - when OTMA first receives an input transaction from XCF, OTMA creates a timestamp to indicate the receiving time of the input transaction. If there is an expiration time specified in the message prefix, OTMA can expire the input.
- Before OTMA en-queues the input transaction, the expiration time is checked and, if the transaction has reached the expiration value, the message is discarded.
- Finally, when the IMS application in the dependent region processes a GU for the message , OTMA once again checks the expiration time and discards the message if it has expired.

Note that for MSC remote transactions, IFP transactions, IMS conversational transactions, and switched-to transaction for program-to-program switches, OTMA only monitors and checks expiration eligibility during the Input Receiving and Enqueuing phases. They cannot be expired during the GU phase.

OTMA Transaction Expiration Processing …

- If IMS detects a mismatch in the OTMA header request
  - NAK is sent to the client using the existing OTMA sense code x'001A'
    - New OTMA reason code:
      - X'0065' - Both TMAMTXP1 and TMAMTXP2 flags are set in the OTMA state data. Only one can be set to request the OTMA transaction expiration function
- What happens when OTMA expires a message?
  - During the Input Receiving Phase or Enqueuing Phase
    - NAK is sent to the client with an OTMA sense code x'0034'
      - OTMA reason code:
        - X'0001' - transaction was cancelled right after OTMA receives it from XCF.
        - X'0002' - transaction was cancelled before OTMA enqueues it.
    - Input message is flagged as expired and discarded

22

When a message is received in IMS through the XCF interface, OTMA checks the validity of the message header. If a mismatch is detected in the flags requesting message level specification, a NAK sense code x'001A' is returned to the client with reason code x'0065' indicating the mismatch.

OTMA issues sense codes that accompany OTMA negative acknowledgement (NAK) messages. For an expired transaction input message condition, the OTMA sense code associated with the NAK is 0034 with an associated reason code of either x'0001' if the transaction was cancelled during the Input Receiving Phase, or x'0002' if the cancellation occurred during the Enqueuing phase.

If the expired transaction condition is detected during the GU phase, an Abend 0243 is issued along with the DFS555I message unless the cancellation occurs in a Shared Queues back-end system in which case a DFS2241I message is issued.

OTMA Transaction Expiration Processing …

- What happens when OTMA expires a message? (cont'd) …
  - During the GU Phase
    - Abend 0243, and
      - DFS555I message, or
      - DFS2241I message when processing in a shared queues back-end
  - In all phases, a 67D0 log record
    - Written when OTMA detects an expired transaction and cancels the transaction

23

OTMA issues sense codes that accompany OTMA negative acknowledgement (NAK) messages. For an expired transaction input message condition, the OTMA sense code associated with the NAK is 0034 with an associated reason code of either x'0001' if the transaction was cancelled during the Input Receiving Phase, or x'0002' if the cancellation occurred during the Enqueuing phase.

If the expired transaction condition is detected during the GU phase, an Abend 0243 is issued along with the DFS555I message unless the cancellation occurs in a Shared Queues back-end system in which case a DFS2241I message is issued.

Transaction Expiration - Considerations

- Order of message expiration value for OTMA messages
  - Use Message Level specification if defined
  - Else, use Transaction Level specification
    - Output Creation Exit – DFSINSX0
    - DRD - CREATE TRAN/TRANDESC, and UPDATE TRAN/TRANDESC
      - Resource Definition Data Set (RDDS) support
    - System definition – TRANSACT macro EXPRTIME
  - The message level of the transaction expiration function is available only if the OTMA member client chooses to implement the capability
    - IMS Connect supports this function

24

For OTMA messages, the expiration value specified on a Message Level is always used if present. Otherwise, the value specified using the Transaction Level methods are honored.

Transaction Expiration - Benefits

- Benefits
  - OTMA environments have the flexibility of using the transaction level expiration support and/or message level overrides
    - Transaction level specification
      - Can be used even without changes in the OTMA Members or member clients
    - Message level specification
      - IMS Connect extends the IRM timer support
        - Application sets a flag, IMS Connect determines the STCK value
        - Facilitates the coordination of timeout values in IMS Connect and OTMA

The ability to discard an input message whose expiration value has been reached provides an automatic mechanism to prevent timed out messages from processing. For OTMA environments, the transaction level specification can be invoked without any changes to OTMA members or remote clients. The OTMA capability for message level overrides, however, provides the additional control on a message by message basis. OTMA members have a choice on how the support is implemented. IMS Connect allows remote clients to simply set a flag to request the expiration value which allows IMS Connect to determine and coordinate the expiration time with other IMS Connect timers.

Enhanced Timeout for Hung TPIPEs

Background Terminology

- OTMA messages and supported interaction are either
  - CM0 (Commit-then-Send)
    - IMS processes the transaction and commits the data before sending a response to the OTMA client (standard IMS model)
      - Replies are queue to a TPIPE (OTMA version of an LTERM)
  - CM1 (Send-then-Commit)
    - IMS commits the transaction output as part of sync-point processing, and then delivers the output to the client later
      - Replies are not queued
- ACK (acknowledgment) /NAK (negative acknowledgment)
  - Response from a remote client required for all CM0 output messages and CM1 messages that are part of synclevel (confirm or syncpoint) processing

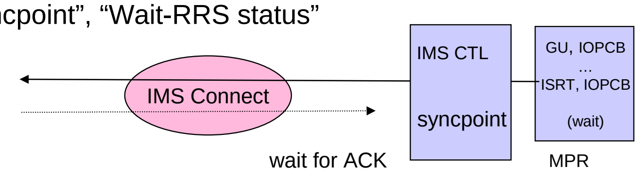
- Hold Queue
  - TPIPE secondary queue used for three types of messages
    - ALTPCB messages chained to the first control block that is used for IOPCB output
    - Send-only messages
    - Rejected IOPCB output
  - Optional for OTMA members
    - IMS Connect is currently the only hold-queue capable OTMA member client

27

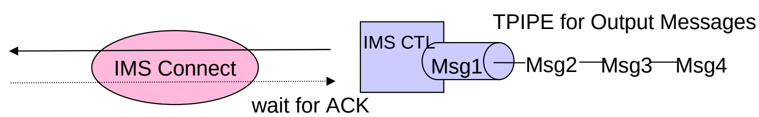
IMS Version 11

Highlights

- IMS 10 introduced Send-then-Commit (CM1) timeout
  - Output in “Wait-syncpoint”, “Wait-RRS status”



- IMS 11 completes the OTMA output message timeout capability
  - New OTMA Commit-then-Send (CM0) ACK timeout enhancement
    - Provides automatic detection of hung TPIPEs
      - Queued CM0 messages requiring an ACK of message receipt



28

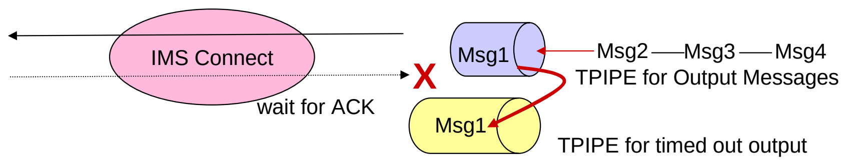
IMS 10 provided a timeout control option applicable to Send-then-Commit (CM1) message processing. As a review, OTMA CM1 response messages associated with synclevel=confirm or synclevel=syncpt requests require an ACK/NAK from the OTMA client. Due to the possibility of a client programming error or a network failure or delay, the expected ACK/NAK may not be received by IMS resulting in IMS holding up resources during the syncpoint process. To resolve this situation, the OTMA CM1 timeout support provided an overridable timeout default of 120 seconds.

IMS 11 completes the OTMA timeout support for output responses with a timeout capability for Commit-then-Send (CM0) messages which stay queued until delivered and acknowledged.

IMS Version 11

Highlights …

- CM0 ACK timeout enhancement
  - Implements timeout action to free the hung TPIPE
    - Moves the CM0 output message causing the hang to a timeout message queue
  - Release the original output queue to allow continued delivery



- Addresses situations
  - When a client application receives a commit-then-send (CM0) output and fails to respond with an acknowledgement (ACK) or a network problem occurs
    - TPIPE in IMS hangs preventing subsequent output on the TPIPE from being delivered

29

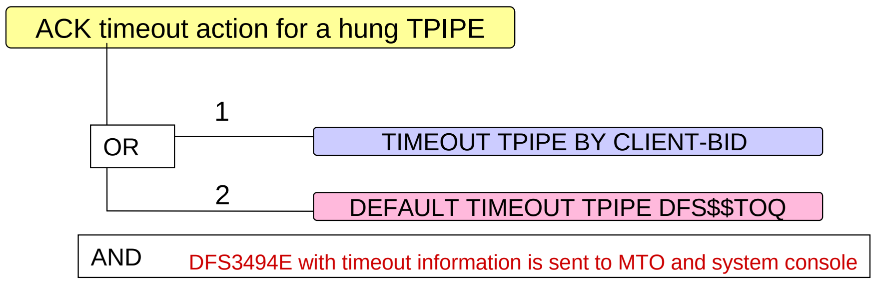
The IMS OTMA CM0 ACK timeout enhancement detects possible output message hung conditions and takes the timeout action so that the CM0 output on the hung TPIPE can be moved to a timeout message queue and the rest of the output on the TPIPE can continue to flow.

The new timeout enhancement address problems in the network or with programming errors on the remote side that result in a non-existent or lost ACK for the CM0 repl.

IMS Version 11

Implementation

- For non-hold queue clients, e.g., all non-IMS Connect members
  - The timed out message is moved as follows:



Client-Bid: process when an OTMA member joins the XCF group and requests a connection to the IMS server

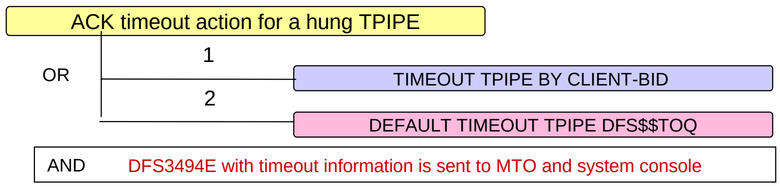
30

For an OTMA client that does not support the hold queue, the timed out CM0 output message is moved to the timeout TPIPE name as specified during the OTMA client-bid time. If a timeout TPIPE name was not provided as part of the client-bid then OTMA moves the message to a default timeout TPIPE named DFS$$TOQ. When a timeout occurs, a DFS3494E with the timeout information is sent to the system console and MTO.

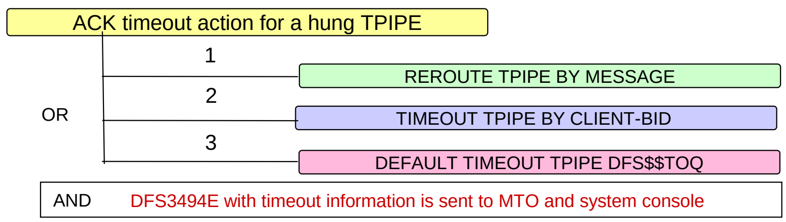
Although WebSphere MQ is a non-hold queue client, this situation is probably not one that would be experienced by MQ environments simply because the MQ IMS Bridge which runs on a z/OS environment is coded to immediately ACK back to IMS on receipt of a message. Additionally, network delays that could lose an ACK are not applicable to this interface between MQ and IMS.

Implementation…

- For hold queue clients, e.g., IMS Connect
  - The timed out message on a hold queue is moved as follows:



  - An IOPCB output message is moved as follows:



31

For an OTMA member client that supports the hold queue, such as IMS Connect, a timed out asynchronous message on the hold queue is sent to the timeout TPIPE name as specified on the client-bid. If no name was specified in the client-bid then OTMA uses the default timeout TPIPE named DFS$$TOQ.

On the other hand, for a timed out CM0 IOPCB output reply, OTMA first checks to see if the message can be rerouted using the reroute TPIPE name specified in the original input CM0 message. If the input CM0 message did not specify a reroute TPIPE name then OTMA follows the path of checking for a name in the client-bid and, if not, then defaulting to the name DFS$$TOQ.

A timeout condition always results in a DFS3494E with the timeout information being sent to the system console and MTO.

Specifying the Timeout TPIPE Name

- Architected OTMA State Data Prefix
  - Flag TMAMSTO EQU X'04' under the existing TMAMHFG2 Client-Bid flag2
    - Indicates that the OTMA client intends to provide the CM0 ACK timeout queue name in the client bid request.
  - 8-byte field TMAMTOQN under the existing TMAMTO timeout value of the state date prefix with the following meaning:
    - This field specifies the 8-byte tpipe name which will be used as the CM0 ACK timeout queue to save the timed out CM0 messages.

OTMA provides the architected interface for member clients to indicate the override timeout queue name.

Specifying the Timeout TPIPE Name – IMS Connect

- IMS Connect implementation – HWSCFGxx parameters
  - HWS statement
    - New parameter CM0ATOQ=cm0atoq
      - 1- to 8-character name to be used for OTMA CM0 ACK Timeout queues
  - DATASTORE statement
    - New parameter CM0ATOQ=cm0atoq
      - 1- to 8-character name to be used for the OTMA CM0 ACK Timeout queue
- Messages that were requeued to the timeout queue
  - Retrieved using RESUME TPIPE request to the timeout queue name
    - Ensure that an ACK is sent
      - Otherwise a timeout will again be triggered

33

IMS Connect takes advantage of the OTMA CM0 timeout support and provides a new parameter CM0ATOQ that can be used to provide a timeout queue name. If specified on the HWS statement, then the name is passed on a client-bid to all datastores associated with this IMS Connect instance. If specified on the DATASTORE statement, then the name overrides the HWS specification and is sent on the client-bid to a specific datastore.

Output messages queued to a hold queue can later be retrieved by a Resume TPIPE command from a client. After a client retrieves an OTMA CM0 output from a hold queue, the client needs to send back an acknowledgement to OTMA. Failing to send the acknowledgement for the hold queue output will again trigger the timeout action.

Timeout Value Specification

- CM0 timeout value – default of 120 seconds
  - Same specification as CM1 timeout (introduced in IMS version 10)
  - Overriden by:
    - DFSYDTx T/O timeout value (0-255)
    - /START TMEMBER TIMEOUT (0-255)
    - OTMA member
      - Client-bid level
      - Remote client message level
- Zero override disables timeout
  - DFSYDTx T/O=0 or /START TMEMBER TIMEOUT 0
  - OTMA detects a possible timeout condition and issues DFS3495W
    - TPIPE remains hung

34

The OTMA CM0 ACK timeout specification leverages the existing timeout specification for CM1 messages that was introduced in IMS 10. The timeout function, therefore for both CM1 and CM0 output reply messages, starts automatically when the IMS system is started, with a default timeout value of 120 seconds. This value can be overriden in several ways:

- /START TMEMBER TIMEOUT command where the timeout value can be 0 to 255
- OTMA descriptor member DFSYDTx T/O timeout
- The OTMA member's client-bid specification, and optionally at an individual message level from an OTMA member's remote client. Overrides provided in this fashion cannot specify 0 nor can they specify a value higher than the OTMA override value as provided in the /STA TMEMBER command or the DFSYDTx descriptor.

Both the /START TMEMBER TIMEOUT command and the OTMA descriptor DFSYDTx can be used to disable the timeout function. The timeout action added in this release for an OTMA CM0 output message is essentially to reroute the timed out message to a different output queue so that the existing output queue can continue to deliver the rest of the queued output messages.

If timeout is disabled, the CM0 message and associated TPIPE remain in hung status waiting for an acknowledgement. OTMA, however, detects this condition and issue a DFS3495W warning message as an indicator that an OTMA TPIPE is waiting for an acknowledgement.

OTMA Messages

- New messages
  - DFS3494E - OTMA HAS TIMED OUT FOR TMEMBER/TPIPE XXXX/YYYY AND MOVED THE OUTPUT TO ZZZZ
    - Indicates that timeout has occurred
  - DFS3495W - OTMA HAS BEEN WAITING FOR AN ACK FROM TMEMBER/TPIPE XXXX/YYYY FOR OVER ZZZZ SECONDS
    - No timeout occurs because CM0 timeout has been disabled
- Existing message
  - DFS1284E - (N)ACK ON NON-WAITING BLOCK IN TPIPE=xxxxxx MEMBER=yyyy TOKEN=zzzzzz
    - Client specified incorrect TPIPE
    - ACK is late and OTMA already timed out the message

35

DFS3494E is issued whenever a CM0 timeout has occurred resulting in a CM0 output message being moved to a different TPIPE. When this condition is detected, the root cause of the missing ACK or NAK for the timeout condition should be investigated. The problem could be a remote application program slowdown or design problem, or even a failure in the network component.

DFS3495W is issued when the CM0 timeout function has been disabled but OTMA detects that a timeout condition is possible because the TPIPE message has been waiting for an acknowledgement. The TPIPE remains in hung status and all of the queued output messages on the TPIPE are not delivered until the ACK or NAK is received.

DFS1284E is an existing message. The explanation has been expanded to state that an OTMA client sent an acknowledgement message, ACK or NAK, to IMS for a transaction pipe (TPIPE) that is not waiting for the message. The client may have specified an incorrect TPIPE name or, alternatively, IMS OTMA may have already timed out the wait.

CM0 ACK Timeout – Migration and Considerations

- Migration
  - New default timeout of 120 seconds
    - Disabling the support requires DFSYDTX specification or /STA TMEMBER
- Considerations
  - Operations
    - Document the new messages
  - IMS systems programmers
    - Be aware of the existence of the timeout TPIPE queue, e.g., DFS$$TOQ
    - Identify the root cause of the missing acknowledgement when the timeout Tpipe queue count is non-zero
  - Application programmers
    - Develop a process to retrieve the timed out message and determine appropriate action

CM0 ACK Timeout - Benefits

- Benefits
  - Ability to keep CM0 messages flowing even when a problem with one message occurs
  - Greater control of an alternate TPIPE name destination for problem messages

Resource Monitoring

Highlights

- Client-server protocol that allows early detection and warning of possible OTMA problems

  – OTMA:
    - Monitors resources - control blocks associated with unprocessed messages
      – Possible flood condition and incomplete Send-then-Commit CM1 messages
    - Detects possible degraded levels
    - Sends messages to clients (OTMA members) about the resources

  – OTMA members, e.g., IMS Connect, to:
    - Stay informed of the status of OTMA resources and problem conditions
    - Support corrective actions such as rerouting the request to another IMS

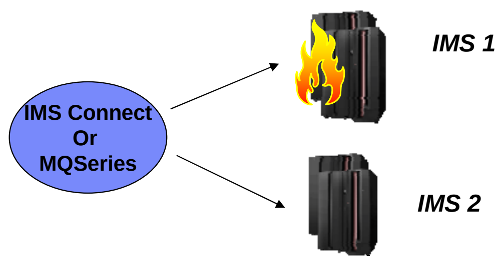
39

IMS 11 OTMA also provides a new client-server protocol that allows OTMA to monitor certain resources used by OTMA functions in order that an automatic detection of possible degraded levels of these resources can occur. If needed, OTMA can process appropriate server actions depending on the condition as well as send out a protocol message to the OTMA Member client. The OTMA member that receives the protocol messages can choose to take corrective action when a degraded condition occurs. One example would be the rerouting of a transaction request from one IMS to another if the target IMS is unable to accept the message. In the situation where none of the other IMS systems are suitable or available to accept a reroute, the OTMA client has the option of marking the IMS system “temporary unavailable, try again later”.

.

Highlights …

- Addresses possible outages
  - When an IMS does not have the resources to effectively process the work submitted by an OTMA member
    - The member continues to send in work leading to a flooded condition



40

The new resource monitoring capability can therefore address the situation when an OTMA client is unaware that a target IMS cannot process the workload and continues sending work to OTMA. This situation can lead to flooded transactions and an IMS outage which disrupts all end client applications.

Background

- OTMA message flood detection and control support from previous releases
  - Monitors the growth of active input messages
    - Default of 5000
    - Overrides
      - DFSYDTx INPUT=
      - /STA TMEMBER INPUT
      - As each OTMA member joins the group
        - The client-bid optionally provides a new INPUT value
  - At 100% of the threshold, new input messages from an OTMA member are rejected
    - Possible delays caused by elongated processing time in IMS
    - Remote environment sends in a flood of messages

In previous releases, OTMA's Message Flood Detection and Control capability provided a mechanism to automatically monitor the growth of active OTMA input messages and the control blocks associated with these requests. Specifically, when an OTMA member or client sends a transaction to IMS, OTMA internally creates a control block called the TIB (Transaction Instance Block) to track each active input message. For a send-then-commit (CM1) message, the control block is used for input and output processing after which the storage is freed or reused. For a commit-then-send (CM0) message, the control block is only used for input processing. If, however, several thousand OTMA input transactions are received and waiting to be processed, thousands of control blocks representing the requests could fill up LSQA storage below the line and possibly cause the IMS system to fail with an S40D abend. To prevent this type of OTMA message flood condition, OTMA suppresses or control the flow of input messages for OTMA based on a maximum value for the number of TIBs allowed for an OTMA member in the system.

Enhancement

- Additionally, IMS 11
  - Provides early warning and detection through
    - Client-server protocol
      - Heartbeat and Action messages
    - Information about control blocks associated with messages as they are received from XCF for each OTMA member client
      - Potential elongated times caused by
      - RACF processing
      - Queue manager processing
      - Internal IMS problems
  - Monitors the total number of active input messages for ***all*** OTMA members

42

The enhanced OTMA flood control support in IMS 11 includes:

- Input message control blocks as documented on the previous visual
- Control blocks associated with input messages as they are first received from XCF
- Total input message control blocks for all OTMA members

Additionally, OTMA provides a set of resource monitoring capabilities that assist in the detection and possible resolution of these resource shortages.

Implementation

- Three-phase protocol
  1. Client-bid phase
     - When the OTMA member connects to the IMS server
       - OTMA initiates an internal monitor to examine the processing resources needed for the client
     - When an OTMA member reconnects to the IMS server
       - Current resource status is sent with the client-bid reply
  2. Processing phase
     - Every 60 seconds - ***heartbeat message*** detailing resource availability
     - Immediately when needed - ***action message*** with information on severe degradation
  3. Client disconnect phase
     - Monitoring continues in anticipation of a reconnect

43

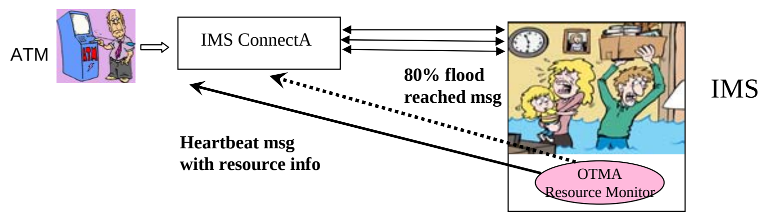
The resource monitoring function occurs in three phases of the OTMA life cycle:

1. When OTMA receives a connection (client-bid) request from a new OTMA client, OTMA then initiates an internal monitor which will be activated every 60 seconds to examine the processing resources needed for this client. If the OTMA client is reconnecting then the current OTMA resource information will be returned as part of the client-bid response.
2. During normal interaction and processing between the OTMA client and OTMA, OTMA maintains a member level availability status code that represents its ability to process work for the OTMA client. This information, a heartbeat message, is sent to the OTMA client every minute in the form of an OTMA protocol message   On the other hand, if the IMS server detects a severe problem, it immediately updates its availability status and sends the status information to the affected OTMA client or member as an action message.. When the condition has been alleviated, the OTMA client is informed as well.
3. After the OTMA client disconnects from the IMS server, OTMA continue to monitor the processing resources so that the resource information continues to be current and available for a subsequent OTMA member/client reconnect.

IMS Version 11

OTMA Resource Monitoring – How would it be used?

- Example 1:
  - IMS ConnectA submits many transactions to IMS
    - OTMA monitoring continually sends a heartbeat message every 60 seconds
  - An OTMA slowdown results in a Message Flood condition
    - An Action message is sent when the threshold reaches 80%
      - Allows IMS ConnectA to stop sending new messages to IMS
    - OTMA continues to send the heartbeat message with the resource info to IMS Connect indicating the warning condition
  - Flood status is considered relieved at 50% of the threshold level



DFS4380W - OTMA XCF MESSAGES FROM membername HAVE REACHED 80% OF THE MAXIMUM MESSAGE LIMIT xxxxxxxx

44

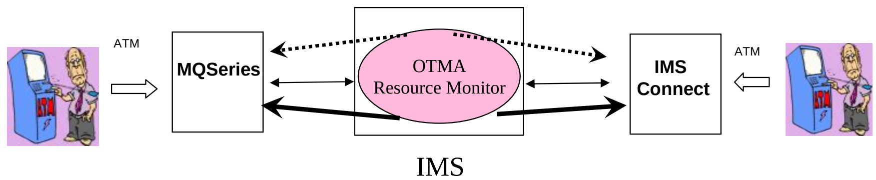
The OTMA resource monitoring capability can be used in several ways. In this first example, an OTMA member client such as IMS Connect receives the heartbeat message every minute. In the case where the IMS resources experience a slowdown, an action message is immediately sent to IMS Connect. IMS Connect exits or vendor products can take advantage of this information and can:

- Do nothing but continue monitoring the resource status
- Stop sending messages to the IMS that has the slowdown
- Reestablish connectivity to IMS when the flood condition is

IMS Connect can monitor both resource availability situations as well as an IMS abend.

OTMA Resource Monitoring – How would it be used? …

- Example 2:
  - Both IMS Connect and MQSeries submit transactions to IMS
    - The total number of un-completed send-then-commit (CM1) transactions reaches the global warning level,
    - OTMA's resource monitor sends action messages to both members
      - Either or both members can choose to react to the messages
    - Heartbeat messages including any flood control status are sent every minute until the condition is relieved



DFS4388W - TOTAL OTMA SEND-THEN-COMMIT(CM1) MESSAGE BLOCKS HAVE REACHED OR EXCEEDED WARNING LIMIT OF xxxx

45

In this second example, OTMA has two member clients. Resource monitoring informs both clients of availability status of the OTMA resources. Heartbeat and action messages are sent to both members and either one or both can choose to react to the information. Both OTMA member clients are therefore informed of availability conditions as well as the severe condition of an abnormal abend of IMS.

OTMA Protocol Message

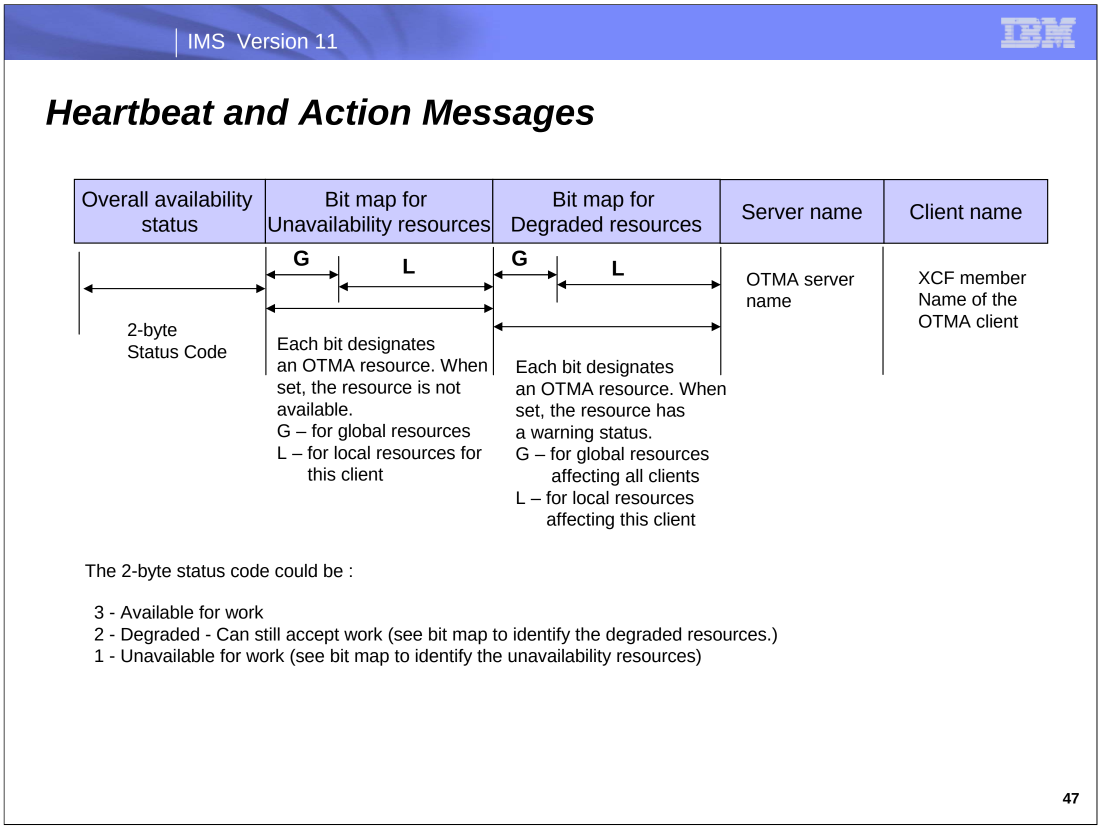
- New command type, located at byte 4 of the OTMA control data prefix
  – X’3C’ informs the OTMA client that OTMA is sending availability information
- OTMA state data prefix contains the following information

Byte	Length	Content
0	2	Length of the state data for protocol message x’3C’
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 or more severe conditions
4	4	Bit map for OTMA resources are in the severe state
8	4	Bit map for OTMA resources are in the warning state
12	1	Status flag - ‘80’ means this is a regular heartbeat message issued every 60 sec 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

46

Availability level information is sent to the OTMA client every minute in the form of the OTMA protocol message with the protocol command type set to X’3C’. In addition to the availability status, the protocol message includes bit maps that identify each supported processing resource classification that could trigger the change in availability.

A status flag, as highlighted in the visual, is set to x’80’ to identify a “heartbeat” message. If the flag is not set, then the message is an “action” message driven by a change in resource availability status.



A closer look at the information in the OTMA state data prefix for both heartbeat and action messages shows how OTMA maintains a member level availability status code representing its ability to process work for an OTMA member. The following status codes in the protocol message are used:

3 – Available for work for this OTMA member; no warning or severe condition

2 – Degraded; can still accept work from this member, but one or more warning conditions exist

1 – Unavailable for work; one or more severe conditions exist

One bit map shows the OTMA resources that are in the severe state, while the other bit map shows the OTMA resources that are in the warning state. These bit maps can be used to determine the cause of the OTMA degraded condition. The server name in the protocol message shows the OTMA server name, and the client name in the protocol message shows the XCF member name of the OTMA client.

Enhanced Global Command

- /STA TMEMBER **ALL** INPUT ####
  - Command enhancement to set the global message flood warning level for **ALL** the OTMA members
    - IMS version 10 introduced the INPUT value
      - Overrides the default of 5000 messages for an OTMA member instance
  - The new **ALL** parameter provides a global value that OTMA additionally monitors for all members combined
    - Default is 8000
    - When the global value is reached
      - WTO, MTO and Auditlog messages
      - Action message is sent to all OTMA members
- /DISPLAY OTMA enhancement
  - New global message flood warning level display under the INPT column of the OTMA Server name

48

When this command is issued, OTMA uses the new specified limit instead of the system default of 8000 to monitor the total active input message count for send-then-commit (CM1) transactions from all of the OTMA members. If the total number reaches this specified limit, OTMA issues a DFS4388W to the IMS MTO and system console along with OTMA protocol messages with the warning status to all of the OTMA clients. When the condition is relieved, OTMA issues a DFS0798I message to IMS MTO and system console along with OTMA protocol messages reflecting a good status to all of the OTMA member clients. The maximum value that can be entered for INPUT is 9999.

Action points

Member	Default Threshold	80%		(at 5% incr) 85%-95%		100%		Relief	
		WTO MTO Auditlog	3C	WTO MTO Auditlog	3C	WTO MTO Auditlog	3C	WTO MTO Auditlog	3C
Individual Member Level	**5000**	X	X	X		X	X	at 50% X	X
Global Level - All Members	**8000**					X	X	at 80% X	X

- Member level
  - At 100% of the member threshold, messages are rejected until level is relieved at 50%
  - /DIS TMEMBER shows a FLOOD condition at 100%
- Global level
  - At 100% of the global threshold, a warning is sent but messages continue to be accepted

The table in this visual shows the percentages at which WTO and X'3C' protocol action messages are issued. The threshold values shown are the defaults of 5000 for an individual member and 8000 for all members globally. These defaults may be changed to more appropriate levels for a specific enterprise.

At the Member Level, both the WTO message and the X'3C' message are sent at 80% as a warning as well as at 100% to indicate a severe condition. At increments of 5% from 85%-95%, only the WTO message is sent. At 50% after the threshold was previously reached, the WTO and X'3C' messages are issued to indicate a relief of the situation.

At the Global Level, the WTO and X'3C' messages are sent both at the 100% mark and subsequently at 80% to indicate a relief of the situation.

OTMA Messages

- New messages sent to WTO, MTO and Auditlog
  - DFS4380W - OTMA XCF MESSAGES FROM membername HAVE REACHED 80% OF THE MAXIMUM MESSAGE LIMIT xxxxxxxx
    - Processing continues
  - DFS4381I - OTMA XCF MESSAGE FLOOD CONDITION HAS BEEN RELIEVED FOR MEMBER membername
  - DFS4388W - TOTAL OTMA SEND-THEN-COMMIT(CM1) MESSAGE BLOCKS HAVE REACHED OR EXCEEDED WARNING LIMIT OF xxxx
    - Processing continues
  - DFS0798I - TOTAL OTMA UNCOMPLETED SEND-THEN-COMMIT(CM1) MESSAGE BLOCKS ARE DECREASING BELOW xxxx
    - Global level is relieved, individual member may still be in a flood condition

50

IMS 10 provided the following messages for the input message flood condition:

- 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.

IMS 11 introduces four new messages:

- DFS4380W is issued when OTMA experiences a slow down due to RACF I/O, queue manager I/O, or other unknown internal hang in the IMS system. Because of this condition, XCF messages from OTMA member client (such as IMS Connect or MQ) could build in IMS. When OTMA detects this condition, OTMA issues this warning message and sends out a protocol message to the OTMA member client so that the client can take action to reroute the subsequent transactions to a different IMS. The XCF messages from the OTMA client will eventually be processed by the IMS when the slow down condition is relieved or resolved. If the OTMA transaction expiration function is activated in the IMS system, these XCF messages could be discarded.
- DFS4381I is issued when the OTMA XCF message count has reached 50% of the maximum limit and is considered a relief of a previous resource problem.
- DFS4388W is issued when OTMA detects the total CM1 message blocks have reached or exceeded the specified warning limit. To OTMA, it is a global warning for all of the OTMA clients.
- DFS0798I is issued when the total uncompleted OTMA CM1 messages from all of the OTMA clients has decreased below the 80% level. Note that although the global OTMA message flood has been resolved, an individual OTMA member may still experience a flood condition.

All these messages are sent to the WTO and the MTO. Additionally, since these are unsolicited messages they are included in the OM audit trail if this has been defined in your environment.

OTMA Messages

- Enhancements to route messages to the MTO/Auditlog
  - Existing messages already routed to WTO
    - 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.
  - New message DFS2386I OTMA is connected to member xxxxxxxx
  - Allows AOI exit to monitor the messages and conditions
  - PK47987/UK28262 (IMS 9), PK49317/UK41633 (IMS 10)

51

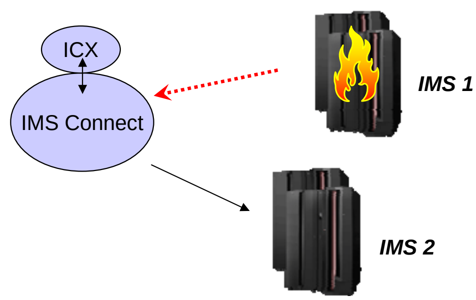
IMS 10 provided the following messages for the input message flood condition that were routed to the WTO.

- 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.

These messages along with a new message DFS2386I are routed to the WTO system console and also to the MTO and Auditlog so that the AOI exit can be written to monitor these messages and associated resource conditions.

IMS Connect Support for OTMA Resource Monitoring

- IMS Connect exploitation of resource monitoring
  - Processes the new protocol messages
  - Updates its data store entry
    - Records new data store events for warning and severe status
  - Provides the information to vendor applications and user exits that can access the information and redirect the transaction requests to a different IMS if needed



IMS Connect exploits the OTMA resource monitoring function by accepting and processing the new protocol messages and updating information in the data store table. IMS Connect does not take action based on the availability status information but rather updates its data store entry so that the IMS Connect user exits or IMS Connect vendor products, such as IMS Connect Extensions, can be enhanced to perform actions such as redirecting subsequent TCP/IP traffic to a different IMS if necessary.

IMS Connect XIBDS update

- Expansion of the "Exit Interface Block Data Store entry" includes:
  - Status flag:
    - x'03' – OTMA server is available
    - x'02' – OTMA server has one or more resources in the warning state
    - x'01' –OTMA server is experiencing some severe resource issue(s)
    - X'00' – No status is available
  - Two 4-byte OTMA server information bit maps that identify which resource is in the severe or warning state
  - A 12-byte timestamp that identifies when OTMA reported this status
    - Updated every 60 seconds when IMS Connect receives the OTMA heartbeat 3C protocol messages
      - Provides additional verification that OTMA is still processing messages

53

The expansion in the "HWSXIBDS - Exit Interface Block Data Store entry" XIBDS includes the following information :

- Status flag:

x'03' – indicates that the OTMA server is available

x'02' – indicates that OTMA server has one or more resources in the warning state

x'01' – indicates that OTMA server is experiencing some severe resource issue(s)

- Two 4-byte OTMA server information bit maps - identify which resource is in the severe or warning state.

- A 12-byte timestamp - identifies when OTMA reported this status.

IMS Connect HWSTECL0 Exit

- Interface that is available to record IMS Connect events
  - E.g., TCP/IP read/write, RACF calls, OTMA send/receive, user exit calls, two-phase commit processing and session errors
    - Allows vendor applications to report on IMS Connect activities and take actions based on any policy defined
  - New event #45 – XIBDS Status Update
    - Based on OTMA protocol message identifying status of OTMA resources

Parameter List Item	Content	Length in Bytes
TOKEN	Token Address	4
EVENT_NUMBER	45	4
EVENT_KEY	EVNT	8
DATA_ADDR_COUNT	1	2
VAR_DATA_LL	0	2
EVENT_DATA_ADDR	Address of XIBDS	4

54

IMS Connect can be customized to facilitate event recording by passing event data to the load module, HWSTECL0. This module stores all trace and event notifications through a recording routine and can be used by any event recording function IMS Connect records events, such as TCP/IP read/write, RACF calls, OTMA send/receive, user exit calls, two-phase commit processing and session errors so that the vendor application can report IMS Connect activities and take actions based on any policy defined.

IMS 11 enhances the capabilities of HWSTECL0 by adding a new event for IMS Connect. Event #45, based on the OTMA protocol message, identifies whether any of the OTMA resources are in a severe, warning, or normal state.

IMS Connect Display Commands

- VIEWHWS, VIEWDS, QUERY MEMBER and QUERY DATASTORE
  - Show the current STATE of the target IMS as reported by OTMA

```
HWSC0001I   DATASTORE=IMS1     STATUS=ACTIVE
HWSC0001I     GROUP=XCFGRP1  MEMBER=HWS1
HWSC0001I     TARGET MEMBER=IMS1          STATE=AVAIL
HWSC0001I     DEFAULT REROUTE NAME=HWS$DEF
HWSC0001I     RACF APPL NAME=ICONALOW
HWSC0001I     OTMA ACEE AGING VALUE=2147483647
HWSC0001I     OTMA ACK TIMEOUT VALUE=120
HWSC0001I     OTMA MAX INPUT MESSAGE=5000
HWSC0001I     SUPER MEMBER NAME=      CM0 ACK TOQ=HWS$DSY
```

The STATE value can be the following:

- AVAIL – indicates that OTMA server is available
- WARN – indicates that OTMA server has one or more resources in the warning state
- SEVERE – indicates that OTMA server is experiencing some severe resource issue(s)
- N/A – indicates OTMA has not reported a status for this Datastore

55

The IMS Connect display commands (VIEWHWS, VIEWDS, QUERY MEMBER and QUERY DATASTORE) have been updated to show the current STATE of the target IMS as reported by OTMA on the Target Member output line. The STATE shows whether the target IMS is available, has one or more resources in the warning state, is experiencing some severe resource issues, or has not reported any status.

OTMA Resource Monitoring – Migration

- Operations and systems personnel
  - New global active input message threshold default of 8000 for all OTMA members
    - DFS4388W message is sent when the 100% max is reached
    - DFS0798I message is sent when the shortage is relieved at 80%
    - /STA TMEMBER ALL INPUT xxxx changes the default threshold
  - Individual OTMA member active input message threshold default of 5000
    - DFS4380W message is sent when the count reaches 80% of threshold
    - DFS4381I message is sent when the shortage is relieved at 50%
    - /STA TMEMBER INPUT xxxx changes the individual member default

OTMA Resource Monitoring - Benefits

- Benefits
  - Allows OTMA member clients to take advantage of early flood detection and failure notification
    - Detect and address a problem when it is starting
      - Reject remote clients from sending in new messages
      - Reroute the messages to another IMS that can process the transactions
    - Note: each member can choose how to take advantage of the capability
      - E.g., IMS Connect provides the information to user message exits and vendor products

57

OTMA member clients can choose to take action such as reject remote clients that try to send in more messages while the resource shortage is in effect or reroute the messages to another IMS that can process the messages.

OTMA Type-2 Commands

OTMA Type-2 Commands

- New Type-2 commands for OTMA to
  - Display information about the OTMA workload
    - Messages in the queue that are received and sent through OTMA
    - Extends the resource monitoring capability by providing a command to analyze potential problems
  - Dynamically add or update OTMA descriptors
    - Allow destination routing descriptors to be in any order
- Issued through the OM API
  - TSO SPOC or IMS Control Center

59

Several new type-2 commands provide greater control over the OTMA environment. Through these enhancements requests from a TSO SPOC or IMS Control Center client can display information relative to the OTMA workload as well as request dynamic changes to the OTMA descriptors.

Query OTMATI

- QUERY OTMATI
  - Requests information on the active workload as represented by a Transaction Instance which can represent
    - A CM0 input message which has not been enqueued
    - A CM1 input message which has not been enqueued
    - A CM1 input message which has been enqueued but which has not executed
    - A CM1 input message which is currently executing
    - A CM1 input message waiting for an ACK for the output message (Synclevel Confirm or Synclevel Syncpoint)
    - An IMS Conversation waiting for the next input message for the conversation.
    - An "orphaned" transaction instance
  - Output shows the information by TMEMBER and TPIPE and can assist in determining possible problems processing the input
    - Length of time that the Transaction Instance has existed
    - Correlation ID of the input message
  - PK88101/UK47069 (adds fields to the base IMS 11 support)

60

The new QUERY OTMATI commands allow the user to monitor the workload in IMS OTMA, specifically the "transaction instances" or messages in the queue that are received and sent through OTMA (internally represented by transaction instance block, TIB). Monitoring the TIB data provides a mechanism to diagnose potential problems that may arise and, correspondingly, to respond proactively in an attempt to avoid or circumvent such problems. Examples of a problem in this area would be excessive storage usage by control blocks associated with each Transaction Instance.

When the command is issued without parameters, e.g., QUERY OTMATI, the information displays the TMEMBER, the TPIPE under the TMEMBER, and the total number of Transaction Instance control blocks which may represent queued messages, continuing IMS Conversations or other problems.

Query OTMATI …

- QUERY OTMATI …
  - Examples:

QUERY OTMATI

TSO SPOC output:

Tmember	TpipeName	MbrName	CC	CCText	MsgCnt
MQ	CSQ81234	IMSA	0	Complete successfully	101
ICONNQ	APPLB	IMSB	0	Complete successfully	275
WAS	APPLC	IMSB	0	Complete successfully	364

QUERY OTMATI MSGAGE(3)

TSO SPOC output:

Tmember	TpipeName	MbrName	CC	CCText	MsgCnt	MessageAge
MQ	CSQ81234	IMSA	0	Complete successfully	3	5
ICONN1	APPLB	IMSA	0	Complete successfully	15	6
WAS	APPLC	IMSB	0	Complete successfully	17	9

QUERY OTMATI MSGAGE(5) SHOW(TRAN)

TSO SPOC output:

Tmember	TpipeName	MbrName	CC	CCText	MsgCnt	MessageAge	Transaction
MQ	CSQ81234	IMSA	0	Complete successfully	2	5	ACCTINQ
MQ	CSQ81234	IMSA	0	Complete successfully	1	7	NEWACCT …
ICONN1	APPLB	IMSB	0	Complete successfully	2	8	INVQRY
ICONN1	APPLB	IMSB	0	Complete successfully	9	6	CHKSTAT
…							
WAS	APPLC	IMSB	0	Complete successfully	4	9	SRCHPART
…							

61

The QUERY OTMATI command has several parameters that can be used when analyzing specific problem situations. The parameters include: TMEMBER, TPIPE, TRANCODE, LTERM, USERID, GRPNAME, MODNAME, CMTMODE, SYNCLVL, MSGAGE and SHOW with subparameters: AGINGVAL, ALL, MSGTKN, CMTMODE, CTTKN, GRPNAME LTERM, MONDAME, MSGAGE, SYNCLVL.

For example, by including MSGAGE(3), the total number of TIBS associated with messages are filtered to a subset whose active time is three seconds or more.

By using other parameters for filtering, like, TRANCODE(…), USERID(…), the display shows a smaller subset of the TIBs (messages).

To display the individual characteristics of each TIB (message) instead of just the total number of TIBS (messages), the SHOW parameter is used, e.g., QUERY OTMATI MSGAGE(5) SHOW(TRANCODE). This format of the command displays the individual TIBs (messages), showing how many transaction codes have an age of five seconds or more. With this facility, the user is able to determine any potential problems in the messaging queue that can be corrected or circumvented.

New Type-2 OTMA Commands

- Four new commands:
  - CREATE | UPDATE | DELETE | QUERY OTMADESC
    - Dynamically modify and query OTMA descriptors
      - Affect destination routing decisions
        - Without having to restart IMS to change the descriptors
      - Log record x'221B' is cut for CRE/UPD/DEL OTMADESC
      - The descriptors are checkpointed as log record x'4035'
    - The required and optional parameters for the descriptors have not changed

62

The OTMA destination routing descriptors were introduced in IMS 10 as enhancements to the DFSYDTx PROCLIB member data set. Changes to the descriptors, however, required a scheduled outage or restart of IMS to take effect. With the new type-2 commands (CREATE OTMADESC, UPDATE OTMADESC, DELETE OTMADESC, QUERY OTMADESC) in IMS version11 changes to these destination routing descriptors can be issued dynamically without any interruption to a running IMS instance.

New Type-2 OTMA Commands …

- Examples – CREATE and UPDATE

CREATE OTMADESC NAME(OTMACL9) TYPE(IMSCON) TMEMBER(HWS1)
SPOC output:

MbrName	OTMADESC	Type	TMember	SMember	CC	CCText
IMSA	OTMACL9	IMSCON	HWS1	N	0	Completed successfully

CREATE OTMADESC NAME(OTMADT2) TYPE(IMSCON) TMEMBER(HWS2) TPIPE(PIPERONE)
SPOC output:

MbrName	OTMADESC	Type	TMember	SMember	Tpipe	CC	CCText
IMSA	OTMADT2	IMSCON	HWS2	N	PIPERONE	0	Completed successfully

UPDATE OTMADESC NAME(OTMACL9) SET(TYPE(NONOTMA))
SPOC output:

MbrName	OTMADESC	Type	CC	CCText
IMSA	OTMACL99	NONOTMA	0	Completed successfully

(changes the destination from IMSCON to NONOTMA which clears any values in the descriptor associated with IMS Connect)

UPDATE OTMADESC NAME(OTMADT2) SET(TPIPE())
SPOC output:

MbrName	OTMADESC	Type	CC	CCText
IMSA	OTMADT2	IMSCON	0	Completed successfully

(clears the TPIPE name – without a specified value the TPIPE name value equals the NAME of the OTMA descriptor)

63

OTMA destination routing descriptors can be created while IMS is up and running. When defined with the CREATE OTMADESC command, new routing descriptors are dynamically included in the running system along with the statically-defined destination routing descriptors in DFSYDTx of IMSVS.PROCLIB. The required and optional parameters are identical to those introduced in IMS V10.

The UPDATE OTMADESC command is used to make modifications to existing destination routing descriptors that were either included in DFSYDTx or added using the CREATE OTMADESC command. Additionally, The Destination Routing Descriptors can change TYPEs from TYPE(IMSCON) to TYPE(NONOTMA) or vice versa. If the TYPE parameter is changed from IMSCON to NONOTMA, the values of the rest of the parameters (TMEMBER, TPIPE, SMEM, ADAPTER, and CONVERTR) are automatically 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 to delete any values for the parameters in question, e.g., coding TPIPE() will clear the value of TPIPE(PIPERONE) for the descriptor and set it to the default of the value in the NAME parameter.

New Type-2 OTMA Commands …

- Examples – DELETE and QUERY
  - OPTION (WILDCARD | NOWILDCARD)

QUERY OTMADESC NAME(OTMACL*) OPTION(WILDCARD)

TSO SPOC output:

MbrName	OTMADESC	Type	TMember	TPipe	SMember	CC	CCText
IMSA	OTMACL99	IMSCON	HWS1	HWS1TP01	N	0	Completed successfully
IMSA	OTMACL*	IMSCON	HWS2		N	0	Completed successfully

DELETE OTMADESC NAME(OTMACL*)

TSO SPOC output:

MbrName	OTMADESC	Type	TMember	SMember	CC	CCText
IMSA	OTMACL*	IMSCON	HWS2	N	0	Completed successfully

(default is OPTION of NOWILDCARD)

DELETE OTMADESC NAME(OTMACL*) OPTION(WILDCARD)

TSO SPOC output:

MbrName	OTMADESC	Type	TMember	TPipe	SMember	CC	CCText
IMSA	OTMACL99	IMSCON	HWS1	HWS1TP01	N	0	Completed successfully
IMSA	OTMACL*	IMSCON	HWS2		N	0	Completed successfully

64

Both the UPDATE and DELETE commands support an OPTION of either WILDCARD or NOWILDCARD. The default value is NOWILDCARD. NOWILDCARD will display/delete (depending on whether the command is QUERY or DELETE) the name with an asterisk as an entry from the table of Destination Routing Descriptors. WILDCARD will display/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 itself is displayed/deleted.

The DELETE OTMADESC command requires only one positional parameter, i.e., NAME. If the NAME parameter contains the masked character of asterisk, it will not delete the group of names it is masking. It will only delete the name with the asterisk as its own entry in the destination routing descriptors. If the OPTION parameter is specified with a value of WildCard, then the group of names under a masked destination routing descriptor name will also be deleted; the name with the asterisk is also deleted.

IMS Version 11

New Type-2 OTMA Commands - Considerations

- Changes to descriptors
  - Persistent across warm and emergency restarts
    - Internal table of destination routing descriptors is automatically rebuilt using checkpoint and log records
  - Cold starts
    - DFSYDTx member has to be manually updated with changes
- XRF alternate, RSR tracking, FDBR regions
  - CREATE, UPDATE, DELETE OTMADESC commands are not valid
    - Checkpoint and log records written on the active IMS are used to update the destination routing descriptors in these environments

65

The destination routing descriptors, whether defined in DFSYDTx or added/updated/deleted using the new type-2 commands, persist whenever IMS is warm or emergency restarted because the table of destination routing descriptors is automatically rebuilt using the checkpoint records (and the log records, in the case of an ERE restart).

Prior to cold starting IMS, any desired changes to the descriptors that may have been added in a previous execution with the dynamic type-2 commands will have to be manually changed in the DFSYDTx member of IMS PROCLIB.

Note that the type-2 CREATE, UPDATE and DELETE OTMADESC commands are not valid in an XRF Alternate, RSR Tracking, nor FDBR Region environment. The checkpoint and log records written as a result of these command executions in the active IMS system will be used to update the destination routing descriptors in these other environments. The commands become valid only when the XRF Alternate or the RSR Tracking environment becomes the active IMS environment.

New Type-2 OTMA Commands - Migration

- When specifying descriptors in the DFSYDTx member
  - IMS V10 required a specific order
    - Most specific to most generic
      e.g., DEST1234
      DEST12*
      DEST*
  - IMS V11
    - Entries can be in any order
    - IMS internally rearranges the order when building the internal table
      e.g., DEST1234
      DEST12*
      DEST*

66

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 come after the group of names the asterisk is masking. For example, following the order from most specific to most generic, the user 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, like, DEST*, DESTT1234, and DEST12*. OTMA will automatically rearrange this internally from most specific to most generic. (NOTE: In V11, the entries in DFSYDTx can be inputted in any order and OTMA will rearrange this internally so that the search order will not have to change. It is still from most specific to most generic.

New Type-2 OTMA Commands - Benefits

- Benefits
  - Dynamic capability to request information and monitor the OTMA Transaction Instances
    - Identify potential problems that may result in outages such as storage shortages
    - Diagnose potential problems that may arise and respond pro-actively to avoid or circumvent such problems
  - Availability of IMS when adding, updating, or deleting descriptors
    - Provides the ability to workload-balance outbound transactions without restarting the IMS.
  - Consistent command interface for OTMA by using TSO SPOC or IMS Control Center

In summary, the five new Type-2 commands for OTMA improve usability by:

- Establishing a mechanism to dynamically monitor the OTMA environment and identify potential problems that may result in outages such as storage filling up with control blocks associated with each Transaction Instance.
- Enhancing the ability to workload-balance outbound transactions without restarting IMS.
- Providing a consistent command interface through the use of the TSO SPOC or the IMS Control Center

Increase in OTMA control block size

- OTMA control block (DFSYTIB) has been increased by 64 bytes to include new fields/values to support the Query OTMATI command
  - userid, groupname, aging value modname, lterm, commit mode, and correlator token
  - The new fields are added to the end of the control block
  - PK88105/UK51053 (IMS 11)

- Migration Consideration
  - Any user modifications or non-IBM vendor software which reference this block must be changed and/or re-assembled to include the new size

68

IMS 11 changes the *size* of the OTMA control block DFSYTIB by 64 bytes (x'40') from the old size of x'110' to the new size: x'150'. Correspondingly the YTIB macro has been changed to recognize the new size. Any user modifications or non-IBM vendor software which references this block must be changed and/or re-assembled to include the new size.

OTMA Enhancements Summary and Overall Benefits

- Greater consistency for Shared Queues environments
  - Error message handling and ALTPCB processing
    - Same functionality in back-end and front-end
- Reduced processing costs and CPU cycles
  - Transaction expiration function and discarding of stale input messages prior to processing
- Increased OTMA resiliency
  - Early detection and ability to address possible OTMA resource problems
    - Resource monitoring with heartbeat and action messages
    - CM0 ACK timeout for hung TPIPEs
- Improved usability and consistency in maintaining OTMA environment
  - OTMA descriptor refresh with type-2 commands

69

In IMS 11, OTMA focuses on 3 important areas:

- Greater consistency in the processing environment for back-end and front-end Shared Queues systems.
- Reduced processing costs and CPU cycles by not processing stale or unwanted transaction messages.
- Increased resiliency when monitor OTMA resources and timing out a hung TPIPE.
- Improved usability by introducing type-2 commands than can refresh OTMA descriptors.

IMS Connect Enhancements
(includes BPE enhancements)

IMS Connect Enhancements

- Functional Enhancements for system reliability, system management, and serviceability
  - IMS Connect configuration member HWSCFGx enhancements
  - Enhanced commands
  - Exit Routine enhancements
    - Port Message Edit Exit
    - HWSTECL0 event record enhancements
    - HWSEXPRM parameter list expansion
    - User Message Exit cleanup
  - Cancel Client ID
  - TCP/IP Auto Reconnect
  - Generated Client ID
  - Performance enhancement
- Usability
  - New Recorder Trace and BPE support

Enhancements for support of IMS Connect API, Synchronous Callout and ODBM are discussed in their respective topics

71

IMS 11 introduces many enhancements that address IMS Connect improvements in system reliability, manageability, serviceability and usability.

HWSCFGx Enhancements

- New parameters to IMS Connect configuration statements
  – HWS
  – TCPIP
  – DATASTORE

- Support enhancements to IMS Connect for
  - CM0 ACK timeout
  - Super member support at a datastore level
  - Inactive socket detection at a PORT level
  - Warning messages and early detection of maximum sockets

72

The IMS Configuration member HWSCFGx provides the mechanism to customize and request functions that are needed by different connectivity requirements.

HWSCFGx …

- HWS statement
  - CM0 ACK timeout support
  - New parameter CM0ATOQ=cm0atoq
    - 1- to 8-character name to be used for OTMA CM0 ACK Timeout queues
    - Overrides the OTMA default DFS$$TOQ queue name
      - Described in the OTMA section
    - Note: existing ACKTO= override timeout value that is already used for CM1 timeouts is also used for CM0 ACK timeout overrides

- DATASTORE statement
  - New parameter CM0ATOQ=cm0atoq
    - 1- to 8-character name to be used for the OTMA CM0 ACK Timeout queue
    - Overrides the HWS value for a specific datastore member

The HWS statement provides a new CMOATOQ value that defines a global override timeout queue name for all the datastores (IMS systems) accessed by an IMS Connect instance. The IMS Connect CMOATOQ value overrides the default name in OTMA. As a reminder, this capability affects output reply messages waiting for an ACK from a remote client.

Additionally, the CMOATOQ parameter is also available at a DATASTORE level to allow specific queue names to be defined for each IMS system.

HWSCFGx …

- DATASTORE statement …
  - Enhancement to the super member function
  - New parameter SMEMBER=smem
    - 1- to 4- character name
      - If invalidly specified, e.g., greater than four characters
        - Existing HWSX0909E message is issued identifying the error
        - IMS Connect initialization is aborted with an AbendU3401 RC04
    - Value of SMEM with '####' disables super member support for a datastore
    - Overrides value specified in the HWS statement

74

The DATASTORE statement has also been enhanced to support a new parameter for super member support.

Prior to IMS 11, IMS Connect support for the OTMA super member feature allowed customers to specify only one super member name for each instance of IMS Connect. When coded on the HWS statement then a default name is provided for all DATASTOREs.

The new SMEMBER parameter on individual DATASTORE statements override the default HWS value. By providing the super member support at a datastore level, IMS Connect is now able to support multiple super member names. A value of '####' disables the super member for a specific datastore. Note that the existing parsing of IMS Connect configuration statements is used. If an invalid SMEMBER value is specified (e.g.,. greater than four characters) then an existing HWSX0909E message is issued identifying the error. If the error is ignored and IMS Connect initialization is attempted, the address space will be aborted with an AbendU3401 RC04.

HWSCFGx …

- TCPIP statement
  - Existing parameter SSLPORT=portid
    - Previously allowed multiple ports to be specified
      - Only one port could be active
    - Now enforces the single SSL port restriction
      - Example: SSLPORT=(8887,8888) results in an abend

```
BPE0003E AN ERROR OCCURRED PARSING PROCLIB MEMBER HWSCF11A
BPE0003E AT LINE 3, CHARACTER 35
BPE0003E FAILING TEXT: "8888)"
BPE0003E UNKNOWN POSITIONAL PARAMETER
HWSX0909E ERROR IN PROCESSING CONFIG MEMBER HWSCF11A; M=XCFG 384
HWSX0909E ERROR PARSING MEMBER; R=68, S=BPEPARSE
BPE0006I HWS  TRSP TCB ABEND U3401-00000004 DIAG=1004000180
…
BPE0006I BPE  JSTP TCB ABEND U4095-00000000 DIAG=1004204144
IEA989I SLIP TRAP ID=X13E MATCHED.  JOBNAME=HWS11A17, ASID=001F.
BPE0006I HWS  USRX TCB ABEND U4095-00000000 DIAG=1004008904
```

    - If multiple SSL ports are required
      - Use AT-TLS SSL
        - Ports are specified as normal ports, e.g., PORTID =
      - Use multiple instances of IMS Connect

75

IMS Connect does not support multiple SSL ports. Previous releases of IMS Connect allowed the specification of more than one SSL port but did not actually support running more than one concurrently. Attempting to actually running multiple SSL ports with multiple sockets with any level of concurrent processing caused a failure in the Language Environment.

IMS 11 restricts users from specifying more than one port in the configuration member. If more than one SSL port is specified on the SSLPORT= parameter, IMS Connect will display error messages to the system console and then abend.

If multiple SSL ports are required, IBM recommends the use of z/OS Application Transparent Transport Layer Security (AT-TLS) instead of IMS Connect's SSL function. When using AT-TLS, the AT-TLS SSL ports have to be specified as normal ports (PORTID= parameter) in IMS Connect's Configuration member. AT-TLS performs the SSL decryption before handing the data to IMS Connect and performs encryption before sending the data to the client.   Refer to *z/OS V1R7.0 Comm Svr: IP Configuration Guide* for details on AT-TLS. An alternative to using AT-TLS for multiple SSL ports is to have multiple instances of IMS Connect with each instance dedicated to using one SSL port.

HWSCFGx …

- TCPIP statement …
  - New parameter PORT=(ID=portid, KEEPAV=nn, EDIT=exitname)
    - ID= defines TCP/IP port number
    - KEEPAV= requests that a signal be sent at predefined intervals (seconds)
      - Keeps a long running session active during periods of inactivity and detects when the partner is no longer available
      - Overrides the TCP/IP stack level value for a particular port
      - Default is 0 which turns off the override and reverts to the TCP/IP stack value
    - EDIT= specifies the new Port Input/Output Edit Exit routine name
      - Default is no exit routine
      - If a specified exit cannot be loaded
        - Port remains closed until OPENPORT or UPDATE PORT command
      - Information on the exit itself is provided in a later section

IMS Connect supports a new PORT statement in addition to the existing PORTID specification. Using the PORT statement, a specific port number can specify a keepalive interval as well as invoke a new exit routine. Sub-parameters are defined as follows:

- ID= A 1- to 8-character decimal field to define the TCP/IP port number. Same rules to the PORTID= parameter apply.

- KEEPAV= A 1- to 8-character decimal field that sets the interval (number of seconds) when a signal is sent in the attempt to keep a long running session active during periods of inactivity and to determine if the partner is no longer available. A value of zero (default) reverts to the use of the value specified in the TCP/IP stack. Non-zero values override the TCP/IP stack which accepts a range from 1 to 2147460.

- EDIT= A 1- to 8-character name of the Port Input/Output Edit Exit load module. The module exit must be coded and placed in the library that is accessible to IMS Connect by JOBLIB, STEPLIB or Linklist. This field is optional and defaults to no exit. If an exit is specified but not available (load fails) it will be reported with existing messages and the port will remain closed. The load will be attempted again when the OPENPORT or UPDATE PORT command is issued.

HWSCFGx …

- TCPIP statement …
  - PORT …
    - Can be specified in addition to the existing PORTID parameter
      - But a port number can only be specified once (PORT, PORTID, SSLPORT)
        - If a LOCAL port needs to be specified, use PORTID
        - If KEEPAV or EDIT needs to be specified, use PORT
    - Duplicate port numbers or invalid specifications of sub-parameters result in an abend at IMS Connect initialization
      - KEEPAV and EDIT do not apply to LOCAL or SSL ports

77

The original PORTID parameter can still be used for port definitions but any ports requiring additional attributes, like the keepalive specification or a request to invoke the Input/Output Edit Exit must use the new PORT parameter.

Note that an individual port number may only be defined once. IMS Connect initialization terminates with an abend if duplicate ports are detected in the PORTID and PORT parameters. Both PORT and PORTID can be specified, however, as long as they reference unique port numbers.

Additionally, “LOCAL” is only valid on PORTID specifications and will result in an initialization abend if coded on a PORT parameter. Likewise, KEEPAV= and EDIT= do not apply to LOCAL or SSL ports.

HWSCFGx …

- TCPIP statement …
  - PORT …
    - Examples:

TCPIP=(PORTID=(9991,9992,9993,LOCAL),SSLPORT=(9998),,,)

TCPIP=(PORTID=(9991,LOCAL),PORT=(ID=9992,KEEPAV=20),PORT=(ID=9993), SSLPORT=(9998),…)

TCPIP=(PORTID=(LOCAL),PORT=(ID=9991,KEEPAV=5,EDIT=CUSTMOD1),PORT =(ID=9992,KEEPAV=20),PORT(ID=9993),SSLPORT=(9998),…)

The examples on this visual show potential mixes of the PORT, PORTID, SSLPORT parameters.

HWSCFGx

- TCPIP statement …
  - New parameter WARNSOC=nn
    - Decimal value between 50 and 99, default of 80
      - If specified value is less than 50
        - Reset to 50
      - If specified value is greater than 99
        - Reset to 99
    - Sets a warning level as a % of the MAXSOC limit
        - Issues new message **HWSS0772W** when warning value is reached
        - Existing message **HWSS0771W** when MAXSOC is actually reached

79

IMS Connect supports between 50 and 65,535 sockets. The maximum number of sockets that an IMS Connect instance supports is specified in the MAXSOC= parameter. When the number of sockets reaches the MAXSOC limit, any new connections are refused and message **HWSS0771W** is issued. IMS Version 11 introduces the WARNSOC and WARNINC parameters as ways to provide early detection of this potential problem. WARNSOC supports a decimal value between 50% and 99% as a warning level.

Note that the following messages can be issued:

- **HWSS0772W THE CURRENT NUMBER OF SOCKETS** n **(**p**%) IS NEARING THE MAXIMUM SOCKETS LIMIT** m**; M=**mc when the socket number reaches the warning level specified in the WARNSOC= parameter of the IMS Connect configuration member

- **HWSS0773I THE CURRENT NUMBER OF SOCKETS** n **(**p**%) IS BELOW THE WARNING LEVEL** w**%; M=**mc when the socket number has decreased below the level specified in the WARNSOC= parameter of the IMS Connect configuration member and has decreased to the reset percentage. The reset percentage is either two times the WARNINC value below the WARNSOC value or 5 percent below the WARNSOC value, whichever is lower.

HWSCFGx

- TCPIP statement …
  - New parameter WARNINC=mm
    - Decimal value between 1 and 49, default of 5
      - If specified value is less than 1
        - Reset to 1
      - If specified value is greater than 49
        - Reset to 49
    - Sets the incremental warning % after WARNSOC value has been reached
        - Again issues HWSS0772W each time sockets increase by incremental value
      - If WARNSOC + WARNINC > 99, default values are used
    - **HWSS0773I** - indicates the number of sockets is below the warning level
      - Reset percentage is the lower of:
        - Two times the WARNINC value below the WARNSOC value
        - 5% below the WARNSOC value

WARNINC provides an additional mechanism for incremental warning percentages.

Example of use: WARNSOC=80, WARNINC=5. When the number of sockets increases to 80% of the MAXSOC value, IMS Connect issues the first HWSS0772W message. When the number of sockets increases to 85%, the second HWSS0772W message is issued and again a third one at 90%. If the number decreases to 89% and then increases to 90% again, IMS Connect does not issue the HWSS0772W to prevent flooding the console with messages. When the number of sockets decreases to the reset percentage of 70%, HWSS0773I is issued and the warning limits are reset. When the number increases to 80%, 85%, etc., again, IMS Connect once again issues the HWSS0772W messages.

HWSCFGx …

- To display the values specified
  - VIEWHWS, VIEWDS or QUERY DATASTORE, QUERY MEMBER
    - Provide information to display the SMEMBER and CM0 ACK TOQ values
      - Additional HWSC0001I message for the DATASTORE enhancements
      - Example

```
R xx,VIEWHWS

       HWSC0001I   HWS ID=HWS1     RACF=Y  PSWDMC=N
       HWSC0001I      MAXSOC=2000  TIMEOUT=6000
       HWSC0001I      RRS=N   STATUS=REGISTERED
       HWSC0001I      VERSION=V11 IP-ADDRESS=009.030.123.148
       HWSC0001I      SUPER MEMBER NAME=IMS1       CM0 ACK TOQ=HWS1TOQ
       HWSC0001I      ADAPTER=N
        ...
       HWSC0001I    DATASTORE=IMS1     STATUS=ACTIVE
       HWSC0001I      GROUP=XCFGRP1  MEMBER=HWS1
       HWSC0001I      TARGET MEMBER=IMS1
       HWSC0001I      DEFAULT REROUTE NAME=HWS$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=IMS2       CM0 ACK TOQ=IMS1TOQ
   ...
```

81

Existing output for IMS Connect commands VIEWHWS and VIEWDS (or QUERY DATASTORE and QUERY MEMBER modify commands) have an additional line to display the SMEMBER and CM0 ACK TOQ values at the DATASTORE level. This is an extra HWSC0001I message.

HWSCFGx …

- To display the values specified …
  - VIEWPORT shows attributes associated with the port

```
R xx, VIEWPORT
    HWSC0001I        PORT=9991     STATUS=ACTIVE KEEPAV=60 EDIT=CUSTMOD1
```

  - Enhancements to VIEWHWS show new socket warning and incremental % definitions along with total number of sockets that are currently active
    - PORT statement shows active number of sockets at the PORT level

```
R xx,VIEWHWS
    HWSC0001I        HWS ID=HWS1     RACF=N  PSWDMC=N
    HWSC0001I        MAXSOC=50  TIMEOUT=5000
               …
    HWSC0001I        WARNSOC=80% WARNINC=5% NUMSOC=4
    HWSC0001I        RRS=N   STATUS=REGISTERED
    HWSC0001I        VERSION=V10 IP-ADDRESS=009.030.123.106
    HWSC0001I        SUPER MEMBER NAME=
              ...
    HWSC0001I        PORT=9999     STATUS=ACTIVE   SOC=3
    …
    HWSC0001I        PORT=9998     STATUS=ACTIVE   SOC=1
```

82

The VIEWHWS, VIEWPORT, QUERY MEMBER and QUERY PORT command output also include any specified values for the keepalive option or the Port Input/Output Edit Exit routine name.

An additional line in the VIEWHWS output shows the WARNSOC and WARNINC values along with the current number of sockets identified by NUMSOC= in the message. The PORT statement also displays a new SOC= parameter which provides the number of active sockets on that port.

HWSCFGx Benefits

- HWS enhancements
  - CM0 ACK timeout support
    - Ability to specify the timeout queue name makes it easier for remote applications to retrieve their messages
      - Otherwise all timed out messages go to a global default queue
- DATASTORE enhancements
  - CM0 ACK timeout support
    - Provides greater granularity
  - SMEMBER support
    - Allows fewer IMS Connect instances with greater flexibility
      - One IMS Connect attaching to all IMS systems can mange and group work according to super member name

83

The IMS PROCLIB member HWSCFGx, provides the mechanism to configure IMS Connect functionality and options.

HWS enhancements:

- CM0 ACK timeout allows remote applications to more easily retrieve messages that were timed out during delivery. Without this support, the timed out messages are sent to a global name of DFS$$TOQ making it more difficult for a specific remote client to only retrieve its messages.

DATASTORE enhancements:

- CM0 ACK timeout at a datastore level provides greater granularity in queue name specifications and the ability to differentiate timeout queues for the various datastores.
- The SMEMBER enhancement allows a single instance of IMS Connect to support different groups of client application programs that use different super members. At a connection level, client applications that send messages to the super member name, rather than the specific IMS Connect name, can be grouped by the super member name that they share. Additionally, fewer instances of IMS Connect can manage multiple super member implementations.

HWSCFGx Benefits …

- TCPIP enhancements
  - Single SSLPORT restriction
    - Detects invalid specification at IMS Connect initialization rather than during processing
  - New PORT parameter
    - Provides a more granular keepalive setting for a particular port versus the stack-wide value
      - Can reduce the time it takes TCP/IP to determine when a socket has terminated ungracefully while not incurring the additional network overhead for all TCP/IP applications
    - Requests the use of the edit exit routine
      - Allows remote clients that cannot conform to the IMS Connect message formats to still use IMS Connect

TCPIP enhancements:

- The single SSLPORT restriction enforces the use of only one SSL port for an IMS Connect instance during initialization. Previous releases experienced problems caused by multiple SSL ports being defined during processing.

- The new PORT parameter KEEPAV enables a more granular setting of the TCP/IP keepalive option versus what is provided at the stack-wide level thereby reducing the time it takes TCP/IP to determine when a particular set of sockets have terminated ungracefully while not incurring the additional network overhead for all TCP/IP applications.

- The PORT parameter also defines the use of a special edit exit routine which allows remote client applications that are unable to conform to the standard message formats to request alteration of messages in conformance with IMS Connect standards.

IMS Connect Enhanced Commands

- SUMMARY enhancement to VIEWHWS and QUERY commands
  - VIEWHWS {**SUMMARY**}
  - QUERY MEMBER TYPE(IMSCON) SHOW(ALL | **SUMMARY**)

- Addition of DATASTORE name on client output
  - VIEWHWS, VIEWPORT, QUERY MEMBER, QUERY PORT

```
HWSC0001I CLIENTID USERID  TRANCODE DATASTORE STATUS SECOND CLNTPORT   IP-ADDRESS
HWSC0001I CLIENT01 USRT001   APOL12   IMS1      CONN      7    1026     009.030.123.148
```

- Benefits
  - Summarized output allows the display to be more readable in situations where there are many sockets
  - DATASTORE name on output makes it easier to determine the IMS system to which the transaction has been routed

85

Although the IMS Connect VIEWHWS (WTOR command) and QUERY MEMBER (z/OS modify command interface) set of commands provide a complete picture of the status of the IMS Connect resources, the resulting display can generate many lines of output (WTOs) that could flood the console particularly if many active sockets have been established. The addition of the SUMMARY option for both these commands allow PORT status and client totals to be displayed but bypass the individual listing of each individual socket for the PORTs. Specification of ALL, which is the default, returns all output fields while specification of SUMMARY excludes detailed client information.

Additionally, the DATASTORE name on the client output for the PORTs has been added to allow easier detection of the IMS to which the transaction has been routed.

Port Message Edit Exit

- New exit routine
  - Allows modification of
    - Input message received from TCP/IP before IMS Connect processing
    - Output message after IMS Connect has formatted the message but before it is sent to the TCP/IP client
  - Specified on the PORT statement in HWSCFGx
    - Allows specialized processing on different ports
      - Called during Open and Disconnect processes of a port
      - Called for every input and output message on that port
    - IMS Connect's answer to the IMS physical edit exit functionality
- Addresses the problem when
  - A remote program cannot conform to the IMS Connect standard message requirements, e.g., IRM header, but needs the functionality of IMS Connect

86

The Port Message Edit Exit provides an exit interface for remote clients that cannot conform to the IMS Connect message format, e.g., pass in a full IRM header, but still needs the functionality of the support. The exit is called for input requests after IMS Connect has received the complete input message from TCP/IP but before any processing has started. On the outbound side, the exit is invoked after IMS Connect has formatted the message but before it is sent to TCP/IP for delivery to the client. The exit routine can change the message structure as needed including increasing or decreasing its size or even canceling its delivery.

As documented in the HWSCFGx enhancement section earlier, the Port Message Edit Exit is defined in the IMS Connect PORT parameter. Specialized exits can therefore be specified on different ports as required by various IMS Connect clients.

Following standard conventions, the exit routine load module must be created with a 1-to-8 character name and, once assembled and bound, must be accessible to IMS Connect through JOBLIB, STEPLIB or Linklist.

Port Message Edit Exit …

- Runs as a BPE exit
  - BPE Type 2 user exit
  - Uses standard BPE user exit parameter list
    - Mapped by BPEUXPL macro
      - Points to HWSEXPIO
- HWSEXPIO
  - Port Message Exit Parameter list

```
HWSEXPIO            DSECT ,     PORT I/O EDIT EXIT PARM LIST
PIOPRM_FUNCTION     DS  CL4     CALL TYPE
*    'INIT' INITIALIZATION,  'READ' INPUT FROM CLIENT
*    'XMIT' OUTPUT TO CLIENT,'TERM' TERMINATION
PIOPRM_FUNC_LVL     DS  X       EXIT FUNCTION LEVEL
PIOPRM_FUNC_BAS     EQU X'01',  BASE FUNCTION
PIOPRM_FLAG1        DS  X       FLAG BYTE
PIOPRM_FLG1UPD      EQU X'80',  MESSAGE WAS UPDATED
PIOPRM_FLG1IPV6     EQU X'40',  IPV6 ENABLED, MAP WITH CLNT_IPV6
PIOPRM_PORT         DS  H       IMS CONNECT PORT NUMBER
PIOPRM_XIB          DS  A       ADDRESS OF EXIT INTERFACE BLOCK
PIOPRM_RETCODE      DS  F       RETURN CODE
PIOPRM_RSNCODE      DS  F       REASON CODE
PIOPRM_END_COMM     DS  0F
* END OF COMMON SECTION -       START OF FUNCTION SPECIFIC PARAMETERS
***********************************************************************
•INITIALIZATION FUNCTION
***********************************************************************
                    ORG PIOPRM_END_COMM
PIOPRM_BUFSIZE      DS  F       BUFFER SIZE REQUIRED FOR UPDATED MESSAGE
                    ORG ,       END OF INITIALIZATION PARMS
***********************************************************************
*
READ AND XMIT FUNCTIONS
***********************************************************************
                    ORG PIOPRM_END_COMM
PIOPRM_ORG_BUF      DS  A       ADDRESS OF ORIGINAL MESSAGE BUFFER
PIOPRM_ORG_SIZE     DS  F       SIZE OF ORIGINAL MESSAGE
PIOPRM_NEW_BUF      DS  A       ADDRESS OF UPDATED MESSAGE BUFFER
PIOPRM_NEW_SIZE     DS  F       SIZE OF UPDATED MESSAGE
*
PIOPRM_CLIENTID     DS  0F      CLIENT ID STRUCTURE
PIOPRM_CLNT_IPV4    DS  0CL8    IPV4 MAPPING
PIOPRM_CLNT_4FMLY   DS  H       , CLIENT FAMILY TYPE
PIOPRM_CLNT_4PORT   DS  H       , CLIENT PORT
PIOPRM_CLNT_4IPA    DS  F       , CLIENT IP ADDRESS
                    ORG PIOPRM_CLIENTID
PIOPRM_CLNT_IPV6    DS  0CL28   IPV6 MAPPING
PIOPRM_CLNT_6LEN    DS  X       , CLIENT SOCKET LENGTH
PIOPRM_CLNT_6FMLY   DS  X       , CLIENT FAMILY TYPE
PIOPRM_CLNT_6PORT   DS  H       , CLIENT PORT
PIOPRM_CLNT_6FLOW   DS  CL4     , CLIENT FLOW INFORMATION
PIOPRM_CLNT_6IPA    DS  CL16    , CLIENT IP ADDRESS
PIOPRM_CLNT_6SCOP   DS  CL4     , CLIENT SCOPE ID
                    ORG ,       END OF READ AND XMIT PARMS
PIOPRM_LEN          EQU *-HWSEXPIO  LENGTH OF HWSEXPIO
```

87

The Port Message Edit Exit runs as a BPE Exit and must conform to that interface. Since it is passed the standard BPE User Exit Parameter List (mapped by the BPEUXPL macro), the exit-type-specific parameter list (UXPL_EXITPLP) can be used to access the Port Message Edit Exit parameter list (HWSEXPIO).

The return codes from the exit are:

- 00 – Processing complete. PIOPRM_FLG1UPD is set for READ or XMIT calls to indicate the buffer has been updated.

- 08 – Cancel message for READ or XMIT calls.

Port Message Edit Exit …

- Note that BPE supports a new exit routine interface
  - Type 1 Exit (previous support)
    - Defined in the BPE Exit List PROCLIB member
      - Processed at address space initialization and during REFRESH USEREXIT command processing
      - Associates a user exit type name with a list of one or more user exit modules
  - Type 2 Exit (new) – used by the Port Message Edit Exit
    - Defined to BPE programmatically at execution time
      - Not specified in the BPE Exit List PROCLIB member
    - Exit type names are generated by the IMS component outside BPE control
      - E.g., IMS Connect defines exit type name PORxxxx for the Port Message Edit Exit name

88

IMS now has two types of BPE user exit routines:

Type 1 exits reference the exit routine interface as provided in the past and are the exits that are defined to BPE in BPE Exit List PROCLIB members. These members are processed at address space initialization, and again when a REFRESH USEREXIT command is issued. Type 1 exit specifications can be changed and updated while a BPE address space is running. They are basically pre-defined by the component using BPE (or BPE itself). For example, the Common Queue Server (CQS) defines a set of exit types (CLNTCONN, INITTERM, etc.) in the internals of the CQS code. The IMS system programmer can define and associate one or more modules with each exit type, but the types themselves are defined by CQS.

With type 2 user exits, the "name" of the exit type is defined dynamically at runtime. Type 2 exit type names are typically generated by the IMS component using BPE based on other specifications outside of BPE's control and are therefore not specified in the a BPE Exist List PROCLIB member. The creation of this exit interface came about because of IMS Connect's need to create an exit type name based on an IMS Connect criteria (in the case of the Port Message Edit exit, the port number and the need to allow for one user exit type per port). IMS Connect did not want to have to define 10,000 static exit names (POR0001, POR0002, POR0003...) to BPE. Instead, the IMS Connect system programmer who defines each port to IMS Connect, can include an optional user exit. As IMS Connect initialization processes each port definition, if the port has an exit associated with it, IMS Connect makes a call to a new BPE service to dynamically create a user exit type (e.g., POR1234 which is generated for port number 1234), and associate a module with it.

Each exit type therefore answers different needs. Type 1 exits are good when a component can define a specific exit name and purpose, e.g., "Initialization exit", "Client Connection exit", etc. Type 2 exits are good for unpredictable things such as requirements that are unknown when the component code, e.g., IMS Connect, is being written, e.g., "Exit for port #12", "Exit for accesses to structure XYZ (vs. structure ABC)", etc.,

Port Message Edit Exit …

- BPE …
  - Main differences in the two exit types have to do with the processing characteristics during a REFRESH USEREXIT command
    - Type 1
      - BPE reprocesses the user exit PROCLIB members
      - Reloads the user exit modules for the types specified on the command
        - Enables updates to user exit definitions without stopping and restarting the address space
    - Type 2
      - Only reloads user exit modules defined to BPE at execution time
        - Since they are not defined externally to BPE, exit definitional values (module names, abend limits) are not changed

89

The main difference in the two exit types deals with what occurs as a result of issuing a REFRESH USEREXIT command. BPE processing is as follows:

For type 1 exits:

- Reads any user exit PROCLIB members that are specified on EXITMBR= statements in the BPE configuration PROCLIB member. Because BPE re-reads these members at the time you issue the command, you can edit the user exit PROCLIB member prior to issuing the REFRESH command and make changes to the user exit definitions. BPE does not re-read the main BPE configuration PROCLIB member, so you cannot change the names of the user exit PROCLIB members, only their contents.
- Loads the user exit modules specified on the EXITDEF= statements for the user exit types specified on the command.

For type 2 exits:

- Loads new copies of the user exit modules associated with the user exit types specified on the command.
- Quiesces all current user exits. This means that the command waits for any active exits to complete processing and delays any new calls to the current exits. This ensures that no user exit is running while the exit is being refreshed.
- Replaces BPE control block pointers to the previous user exit modules with pointers to the newly loaded modules. These pointers are used to manage the calling of the exits.
- Resumes the user exits and enables calls to be made to the newly-loaded exits.
- Deletes the old copy of the user exits.

Port Message Edit Exit …

- Managed by BPE commands
  - DISPLAY USEREXIT and REFRESH USEREXIT
    - Exit type name is PORxxxx where xxxx is the port number
      - Examples

```
PORT=(ID=9991,KEEPAV=5,EDIT=CUSTMOD1)

F IMSCONN1, DISPLAY USEREXIT NAME(POR9991)
 BPE0030I   EXITTYPE   MODULE      OWNER    ACTIVE    ABENDS
 BPE0000I   POR9991    CUSTMOD1    HWS         1         0
 BPE0032I   DISPLAY USEREXIT COMMAND COMPLETED

F IMSCONN1, REF USRX NAME(POR9991)
 BPE0032I   REF USRX COMMAND COMPLETED
```

- Benefits
  - Provides an interface to assist remote clients that cannot conform to the IMS Connect standard message layouts

As a BPE exit, both the DISPLAY USEREXIT and REFRESH USEREXIT commands can be used. This visual gives an example of the commands issued against an exit defined against PORT 9991.

The Port Message Edit Exit, therefore, can be of benefit to client applications that need the functionality of IMS connect but are unable to conform to the standard message formats and, in many cases, have their own message format that must be honored.

HWSTECL0 Enhancements

- HWSTECL0 – continues to support event recording functions
  - Adds support for new event records that report activity of the Port Message Edit Exit

Event number	Event key	Event Description
89	SVT Token	Port Message Edit Exit entered
90	SVT Token	Port Message Edit Exit returned
46	Event	Port Message Edit exit initialization
47	Event	Port Message Edit exit returned,

- Benefit
  - Provides information about the new exit's activity that can be used by monitoring tools such as IMS Connect Extensions

IMS Connect facilitates event recording by passing event data to the load module, HWSTECL0. This module stores all trace and event notifications through a recording routine and can be used by any event recording function. IMS 11 produces four new event records for HWSTECL0 that report the activity through the Port Message Edit Exit.

HWSEXPRM Parameter List Expansion

- Expansion of the existing HWSEXPRM macro that is used by IMS Connect user message exit routines

Section	Old Size (bytes)	New Size (bytes)	Comments
Common	12	12	No change
EXPRM (input)	60	100	Add 40 reserved bytes at end
EXRET (output)	40	60	Add 20 reserved bytes at end
EXEXTN	16	40	Add 24 reserved bytes at end
EXPRM_XID	140	140	No change
Total size:	268	352	

  - Note - re-assemble IMS Connect exits
- Benefits
  - Accommodation of future user requirements
  - Improved readability and serviceability of the parameter list
    - Reorganization of the current fields for better placement
    - Elimination of discrepancies caused by lack of space

92

The expansion of the HWSEXPRM parameter list provides the following benefits: accommodation of future user requirements; reorganization of the current fields in the parameter list such that they appear in their appropriate input or output section; elimination of discrepancies caused by unaffiliated field placement due to lack of space; and provision for general clean up to make the parameter list more readable and serviceable.

The changes to the HWSEXPRM macro affect the IMS user message exit routines which have to be reassembled due to the expanded parameter list. System programmers who modify these exits should be aware of the following information:

- Each functional segment in the message exit routines (INIT, READ, XMIT, TERM, EXER, and RXML) has an input and output section with a size equivalent to EXPRM and EXRET respectively. The expansion of the parameter list most directly affects the READ function which has already exceeded the current capacity of the EXPRM input section.
- The EXPREA_SVT and EXPREA_LSTNPORT fields have been moved from the previous location in the READ function EXRET output section back to their rightful location in the READ function EXPRM input section.
- The length of the IPv4 and IPv6 read input areas have been standardized and increased to 32 bytes to allow for future expansion.
- Documentation has also be included in the HWSEXPRM macro to reinforce the notion that the two areas overlay one another.
- Safety checks have been added to ensure that each functional section remains within the bounds of the primary sections.
- A version number field has added to the common section of the parameter list.

User Message Exit Cleanup

- Removal of HWSIMSO0 and HWSIMSO1 Message User Exits
  - No longer shipped with IMS Connect
  - HWSX0908W is issued if the old exits continue to be specified in the IMS Connect configuration member
  - Previous releases documented a recommendation to move from these exits to the newer exits
    - HWSSMPL0/HWSSMPL1 provide enhanced functionality and are delivered as source code
- Benefit
  - Reduces the number of user message exits that have to be maintained

93

Previous IMS Connect documentation recommended that IMS Connect Roll-Your-Own (RYO) applications use either the HWSSMPL0 or HWSSMPL1 User Message Exits. These exits provide enhanced function over the original but functionally stabilized, non-modifiable OCO exits HWSIMSO0 and HWSIMSO1. Additionally, the newer exits are shipped as source code to allow for customer modification. As a result, the older user message exits are no longer needed and have been removed from IMS 11.

User-Defined Messages

- Enhancement to the Message Exit interface
  - Allows an exit-defined user message to be sent in reply to an input message
    - Message can be up from 1 - 128 characters
    - Supports error conditions
      - Detected by IMS Connect or the message exit
        - E.g., allows the exit to perform upfront editing and reply with a user-friendly message in the case of a detected error condition
  - Along with a request to keep the persistent socket connected
  - Implemented in HWSJAVA0, HWSSMPL0, HWSSMPL1

94

IMS Connect enhances the message exit interfaces for HWSJAVA0, HWSSMPL0, and HWSSMPL1 to support sending a user-message (created by the exit) to the client application program as a reply to an input message without terminating the persistent socket connection. The socket, if it is maintained, can be used for the next client interaction.

Users who wish to utilize the user-defined message function need to modify the user message exits HWSSMPL0, HWSSMPL1, or HWSJAVA0 to build the appropriated message with the message length range from a minimum of 1 to the maximum of 128 characters. If the specified message length is greater than 128, the message will be truncated to 128.

IMS Connect requires the buffer space for the message to be at least 5 characters (4 for llzz field and 1 for minimum-length message), otherwise the message will not be built.

User-Defined Messages …

- HWSJAVA0
  - Set a RC of 20 (x’14’) in EXPREA_RETCODE
    - Requests IMS Connect to determine whether or not to keep the persistent connection
  - Set RC of 48 (x’30’) in OMUSR_RETCODE and reason code (RSN) ICONSUCC in OMUSR_RSNCODE of the OTMA header
    - Requests that a user-defined message be sent back to the client
      - If RSN other than ICONSUCC is set, only the RC and RSN are sent back to the client application
  - Provide a user-defined message in the data portion of the OTMA header.
- HWSSMPL0/HWSSMPL1
  - Set a RC of 20 (x’14’) in EXPREA_RETCODE to request IMS Connect to determine whether or not to keep the persistent connection open
  - Provide a user-defined message in the output message buffer

95

To take advantage of this capability for HWSJAVA0 (e.g. for IMS TM Resource Adapter users), exit changes include the following actions:

- Set a RC of 20 (x’14’) in EXPREA_RETCODE to request IMS Connect to determine whether or not to keep the persistent connection. This depends on the socket status after sending the user defined message back to the client application.
- Set a combination of a RC of 48 (x’30’) in OMUSR_RETCODE and a reason code (RSN) of ICONSUCC in OMUSR_RSNCODE of the OTMA header to request that a user-defined message be sent back to the client application. If an RSN other than ICONSUCC is set, only the RC and RSN are sent back to the client application.
- Provide a user-defined message in the application data portion of the OTMA header.

To take advantage of this capability in HWSSMPL0 or HWSSMPL1, the exit changes include the following actions:

- Set a RC of 20 (x’14’) in EXPREA_RETCODE to request IMS Connect to determine whether or not to keep the persistent connection open. This depends on the socket status after sending the user-defined message back to the client application.
- Provide a user-defined message in the output message buffer.

User-defined Messages …

- HWSIMSCB
  - Mapping macro for the user-defined message

```
*****************************************************************************************
*  USER DEFINED MESSAGE SEGMENT
*****************************************************************************************
UDMMASK        DSECT                  USER DEFINED MESSAGE DSECT
UDM_LEN        DS    H                LENGTH OF UDM
UDM_RSV        DS    H                RESERVED
UDM_TEXT       DS    0C               MESSAGE TEXT STARTS
UDMMASK_LEN    EQU   *-UDMMASK        SIZE OF UDM HEADER
```

96

The IMS Connect HWSIMSCB macro defines a mask for the user-defined message as shown on this visual.

User-Defined Messages …

- Impact
  - Modify existing client application to handle the new message
- Benefit
  - Allows a user-friendly message to be sent in error situations
  - Maintains persistent socket connections for user message sendback type flows
    - Prevents disconnection after error condition
      - Reduces the overhead for a client having to reestablish the persistent connection for the next interaction

97

If the user exit takes advantage of this enhancement and sends back a user-defined message, the remote application needs to be able to handle the new message.

Sending a user-defined message makes it easier for a remote client application to understand an error condition detected in IMS Connect. Additionally, if a persistent socket is being used, the exit routine capability to keep the socket open reduces the overhead of having to reestablish the connection.

HWSP1410W Enhancement

- Message HWSP1410W Enhancement
  - Provides additional error information when IMS Connect has problems freeing storage
    - Address of storage in error
      - Existing messages already includes error return code, type of storage and the module that encountered the error
- Benefit
  - Greater assistance in problem determination
    - Enhances message content for problem determination

98

IMS Connect reports errors when freeing storage with message HWSP1410W. This message reports the error return code, type of storage and the module that encountered the error. To further aid problem determination, the address of the storage is being added to this message.

HWSP1410W FAILED TO RELEASE STORAGE; R=*rc*, B=*bn*, A=*a*, M=*mc*

Explanation:

IMS Connect is unable to release storage for an internal buffer.

In the message text:

rc	identifies the return code for the failure to release storage
bn	identifies the type of buffer
a	identifies the address of the buffer (this is what is being added)
mc	identifies the module issuing the message

Note that the associate error is reported in the BPETRACE by BPE services with all the appropriate diagnostic information.

Cancel Client ID Support

- Ability to re-establish connectivity using an existing client ID
  - Automatically cleans up a previous active connection if necessary
    - With support for all interaction types – CM0, CM1, Persistent socket, Transaction socket
  - Bypasses complexity of the existing “cancel timer” alternative
  - PK70327/UK40632 (IMS 9), PK73829 (IMS10)
- Addresses error conditions where
  - A remote client experiences a connection failure
    - But IMS Connect detection of the original failure is delayed
      - Network delays, keepalive, etc.,
    - Attempted client reconnect fails due to a duplicate client condition

The connection between a remote client application and IMS Connect sometimes gets disrupted due to TCP/IP failures, processing failures on the client side, etc. Occasionally, this creates a problem when IMS Connect does not immediately detect the connection break and the remote application attempts a reconnection. Because the original client status may still be active in IMS Connect, a request to reestablish the connection to the same port using the same clientid can result in a duplicate client condition and message HWSS0742W.

With the new function, the client application can establish a new connection with a request to “Cancel Client ID” which causes IMS Connect to automatically discard and cleanup any previously active session for that client ID. All CM0 or CM1 transactions running on either a persistent or transaction socket are supported

Cancel Client Id Support …

- Implementation
  - Cancel Client ID is an option which can be set in the first input message
    - IRM_F3
      - IRM_CANCID EQU X'80'
    - If a duplicate session exists, it is unconditionally cleaned up before the new one is established
      - HWSS0743W is issued
  - IMS Connect exits
    - HWSSMPL0, HWSSML1, and HWSSOAP1
      - Can set the Cancel Client ID flag
      - Can receive the RSM indicator that a client id was cancelled

100

A Client application that is establishing a session with IMS Connect can request "Cancel Client ID" by setting an indicator in the IMS Request Message (IRM) header.

When IMS Connect has determined that the request involves canceling the old client id, the RSM return code is x'30'. This return code is labeled as follows in the HWSIMSCB:

RSMRCT_TMOUT5 EQU X'00000030' CANCEL CLIENT ID

The Return Code and Reason Code are sent back to the client application as follows:

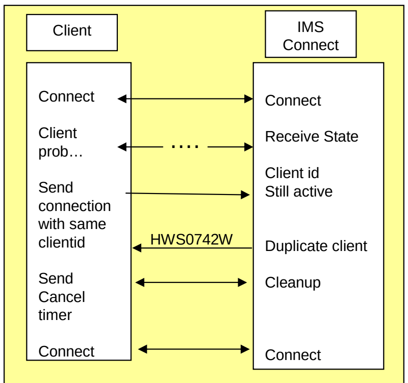
Return Code = x'08'
Reason Code = 'CNCLIENT'' with a corresponding value as x'60'
(RE: LBL=RSTABL in HWSSMPL1)

IMS Connect exits: HWSSMPL0, HWSSML1, and HWSSOAP1 can set the IRM_CANCID flag as well as view the RSM return code.

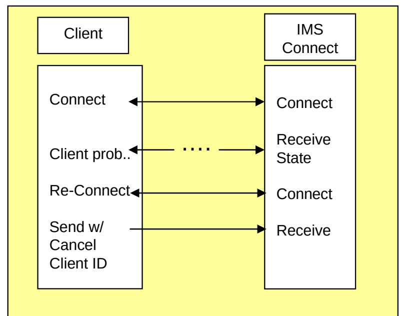
IMS Version 11

Cancel Client Id Support - Benefits

- Benefits
  - Faster, automated way to reestablish a failed connection
    - Cancels an existing socket connection and establishes a new socket connection with the same Client ID
    - Simpler than the cancel timer alternative



Cancel Timer solution (previous)



Cancel Client ID solution (new)

101

The new "Cancel Client ID" support provides a faster, automated way to cleanup hung connections than previous solutions. This capability is much simpler than the alternative that was previously introduced with the "cancel timer" support which required that the remote application be coded to recognize the duplicate client HWS0742W message, issue a Cancel Timer request in the IRM to clean-up of the old session, then turn around and establish a new session.

Generated Client ID

- Mechanism to request that IMS Connect generate a Client ID
  - Impacts ITMRA (IMS TM Resource Adapter) environments
    - For shareable persistent sockets
      - Where the client ID representing a unique socket/TPIPE must be generated rather than end-user specified
- Addresses duplicate client ID error
  - When multiple WAS instances are configured
    - Each ITMRA generates a unique client ID which may or may not be unique across multiple WAS instances

102

The IMS TM Resource Adapter (ITMRA) supports both dedicated and shareable persistent sockets. Dedicated persistent sockets only support CM0 (commit mode 0) interactions and require user-specified Client IDs that represent the socket connection and TPIPE. Shareable persistent sockets, on the other hand, can be shared (serially reused) by multiple applications running either CM1 (commit mode 1) or CM0 interactions.

For shareable persistent sockets, ITMRA does not allow the remote client to specify the client ID, rather it uses an algorithm to generate one for this type of interaction. Although the algorithm is coded to ensure that the value is unique within an ITMRA instance, the possibility exists for the same value to be generated by ITMRA across multiple WAS instances. If this happens then a request from ITMRA could be rejected with a duplicate client error status. To prevent this error condition, environments that have many instances of WAS running on distributed platforms use different IMS Connect ports. Such a setup requires the administration of the WAS configurations and many additional TCP/IP ports for IMS Connect.

IMS Version 11

Generated Client ID …

- Implementation details – provides two options
  - Send Client ID field with blanks on initial establishment of the socket
    - IMS Connect generates client ID
  - New: Send Client ID field with a generated value and set flag OMUSR_F2_CIDREQ in byte OMUSR_FLAG2
    - If the value in the Client ID field is a duplicate
      - IMS Connect generates another client ID
        - Returns flag OMUSR_F2_CIDGEN in byte OMUSR_FLAG2 to indicate a generated clientid is being returned
- Benefit
  - Eliminates the requirement for different IMS Connect PORTs for instances of distributed WAS using ITMRA Shareable Persistent sockets
    - Simplifies WAS configuration and operations management

103

To address the situation, IMS Connect provides mechanisms that generate the Client ID for ITMRA shareable persistent sockets and ensures uniqueness so that all the instances of WebSphere can specify the same IMS Connect TCP/IP port.

The IMS Connect protocol provides two options:

- The first option is for ITMRA to send a blank Client ID. The blank value tells IMS Connect to generate a unique Client ID to represent the shareable persistent socket.
- The second option allows ITMRA to generate a Client ID, as it has in the past, but with a flag setting of OMUSR_F2_CIDREQ in byte OMUSR_FLAG2. The flag tells IMS Connect to generate a unique clientid only when the provided clientid results in a duplicate client condition. In this second case, IMS Connect returns the newly generated Client ID to ITMRA on a reply message.

The generated Client ID support, therefore, ensures Client ID uniqueness (socket/TPIPE name on the IMS Connect/OTMA side) for each shareable persistent socket so that all the instances of WebSphere can specify the same IMS Connect TCP/IP port.

TCP/IP Auto Reconnect

- Automatic reconnection to TCP/IP when network becomes available after a failure
  - New loop and timer logic on each active port
    - Invoked at the end of the terminate port thread process
      - Only if termination was due to failure and not to a command
    - Internally issues an OPENPORT
      - Loops back to timer delay if network is still unavailable
      - Issues HWSS0780I message for each port that is successfully activated

- Benefit
  - Eliminates the need for operator intervention, e.g., OPENPORT command
    - Minimizes recovery time for connections after a network outage

104

If the TCP/IP network experiences a disruption during IMS Connect processing, operators must issue the OPENPORT command when the network is once again available. With the TCP/IP Auto Reconnect support in IMS 11, the same situation no longer requires operator intervention because IMS Connect continues to listen on the socket associated with each active port and automatically attempts to reestablish a connection as appropriate .

The new function includes loop and timer logic that is invoked at the end of the terminate port thread process during port cleanup. It is activated for each port that was active during the time that the connection to the TCP/IP network failed but will not be invoked if the connection to the port was terminated by a command. When a network failure occurs, timers are activated on each port that was active. After a small delay, the threads associated with each of the ports are posted and dispatched to perform an internal OPENPORT command. If the network resources are not available during the retry, the logic once again falls into the small timer wait and a repeat of the process occurs. If the network resources become available, a successful retry results in message HWSS0780I being issued for each successfully opened port.

Performance Enhancement

- New hashing mechanism for client ids
  - Each port has its own hash table
  - PK57574/UK42318 (IMS 10)

- Benefit
  - Reduces CPU overhead
  - Increases throughput
  - Note: improvement varies based on environment

105

A new hashing mechanism for client ids enhances IMS Connect performance by reducing CPU overhead and increasing throughput.

IMS Connect Recorder Trace
and
BPE Enhancements

New IMS Connect Recorder Trace

- Capability that allows the IMS Connect Recorder trace to exploit the new BPE Direct External Trace facility
  - Routes Recorder Trace data to the BPE External Trace data set
- Addresses the limitations and restrictions of the current Recorder Trace data set facility
  - Inability to continuously and reliably capture Recorder Trace diagnostic data without interruption, manual intervention, or data loss due to data set full conditions.

107

IMS 11 introduces a new tracing capability for IMS Connect that takes advantage of the BPE external trace data set support. This enhancement address the problems associated with the limitations of the existing capability which is based on a dataset defined to the IMS Connect startup JCL.

Background

- IMS Connect Recorder Trace data in prior releases
  - Written to a separate, fixed block, fixed LRECL
    - HWSRCORD DD statement in the IMS Connect startup procedure.
      - Precludes dynamically modifying or changing the physical data set without bringing down the IMS Connect address space.
  - Fixed characteristics of the data set limit the type and amount of data captured
  - Access method used to manage the data set
    - Prevents high volume queued I/O activities, easy detection and reaction to end of data set conditions, and the use of multiple data sets or data set switching

108

For many releases, IMS Connect has supported a Recorder Trace capability which writes trace data to a fixed block, fixed LRECL data set that is defined in the HSWRCORD DD statement of the IMS Connect startup JCL. Specifying the data set via a DD in the startup procedure precludes dynamically modifying or changing the physical dataset without bringing down the IMS Connect address space. Finally, the access method employed to manage the data set prevents high volume queued I/O activities, easy detection and reaction to end of data set conditions, and the use of multiple data sets or data set switching.

Background …

- As a reminder, the BPE External Trace data set facility added in IMS 10 addressed the limitations and restrictions described for the current Recorder Trace data set:
  - Variable block with variable length records.
  - Block size chosen by user and LRECL based on BLKSIZE
  - Dynamically allocated based on EXTTRC parameter in BPECFG PROCLIB member.
  - Dynamically re-configurable via the BPE UPDATE TRACTABLE command using the OPTION(REREAD) facility.
  - Designed to handle high volume queued I/O activities and to detect end of data set condition and automatically switch to new data set.

Many limitations of the IMS Recorder data set can be addressed through the use of the BPE External Trace data set facility that was added in IMS 10. For this reason, IMS Connect has chosen to take advantage of the BPE functionality.

Background …

BUT … BPE still has some issues

- BPE trace data is written as an entry in a designated in-core trace table
- Each trace table entry has a fixed length that can range from 16 to 256 bytes (by powers of two)
- The trace table entry length is predetermined - defined when the trace table is created
- The BPE External Trace facility copies the table entries to a copy buffer which is then passed to an AWE processor and written to the BPE External Trace dataset
- The copy process is asynchronous which could lead to data loss if the copy buffer process is overloaded or delayed

… as does IMS Connect

- Each Recorder Trace entry is first captured in a USTAT block with a length of 1440 bytes
  - It would be useful to be able to dynamically adjust the length of the data as needed
- After the data has been captured, the USTAT block is queued to an FWE processor and written as an individual record to the Recorder Trace dataset
- Buffering is efficient and writing individual records does not lose data – unless the data set fills

110

As initially delivered in IMS 10, however, the BPE external trace facility still has some considerations.

BPE trace data is stored in an entry in a designated in-core trace table with each trace table entry supporting a predetermined fixed length which can range from 16 bytes to 256 bytes incremented by the powers of two. The BPE External Trace facility asynchronously copies these table entries to a copy buffer which is then passed to the BPE External Trace I/O AWE Processor and written to the BPE External Trace data set. The asynchronous nature of the copy process can result in the loss of trace data during high volume tracing situations when a trace table wraps and the copy process falls behind and does not capture the data before it is overwritten.

SO …. A Two Part Solution

- BPE Trace Enhancements
  - The first part, or foundational layer, is an extension to the BPE External Trace facilities delivered in IMS 10 and provides:
    - BPE Variable Trace
    - BPE Immediate External Trace
    - BPE Direct External Trace
    - BPE Immediate External Write
- Recorder Trace Enhancement
  - The second part, or implementation layer, is IMS Connect's exploitation of BPE Direct External Trace to re-route Recorder Trace data to the BPE External Trace

The Recorder Trace enhancement, therefore, requires a two part implementation. The first, the foundational layer, is an extension to the BPE External Trace facilities delivered in IMS 10. The extension, called BPE Direct External Trace, allows BPE, or any product running on top of BPE, to write ad hoc data directly to the BPE External Trace dataset.

The second, or implementation layer, is IMS Connect's exploitation of BPE Direct External Trace to re-route Recorder Trace data to the BPE External Trace dataset.

BPE Trace Enhancements

- BPE Trace Facility
  - Continues to support fixed trace table entries
  - Characteristics
    - Fixed table entries have the following format
      - The first two bytes of the entry are designated as the entry code and subcode.
      - The last two words of the entry contain the STCK value at the time the entry was made.
      - The bytes in between are available for user data.
    - Fixed trace table entries have a length that is a power of two between 16 and 256 bytes.

```
 0    1    2    3    4                                         LL-8            LL-1
 |  CD  |  SC  |  B1  |  B2  |  <<<<<<<<<<    user data    >>>>>>>>>>>  |    8-byte STCK    |
```

```
Where:
    LL   =   Length of trace entry (in bytes).
    CD   =   Trace code - Each component is assigned one or more trace codes.
    SC   =   Trace subcode - For each trace code a component may define up to 256 trace subcodes.
    B1   =   User byte 1 - A byte available for user trace data.
    B2   =   User byte 2 - A byte available for user trace data.
    data =   Any data the user wants to put in the trace entry.
    STCK =  Store Clock value at the time the trace entry was obtained.
```

112

The BPE trace facility continues to support fixed trace table entries that have a length that is a power of two between 16 and 256 bytes. The structure of these entries is shown on the visual.

BPE Trace Enhancements

- BPE Trace Facility …
  - Provides a new Variable Trace Facility
  - Characteristics
    - Variable trace table entries have the following general format:
      - Prefix section
      - Fixed entry section
      - Variable data section

```
             4         8         12
0    ----------------------------------------
     -  LLZZ  -  VLEN  -  REC#  -  CONT     -
16   ----------------------------------------
     -                                      -
     -             Fixed entry              -
     -    length is set in THE_ENTLEN       -
     -      (length = 16 | 32 | 64)         -
     -                                      -
nn   ----------------------------------------
     -                                      -
     -            Variable data             -
     -     (offset = 16 + TTHE_ENTLEN)      -
     -           (length = VLEN)            -
     -                                      -
     ----------------------------------------
```

```
Where:
     LL   =  Trace entry length (in bytes)
     ZZ   =  Previous entry length (in bytes)
     VLEN =  Trace entry variable data length
     REC# =  Trace entry variable record number
     CONT =  Trace entry continued record number
     16   =  Prefix length
     nn   =  Offset to variable data section
```


Additionally, the BPE trace facility has been enhanced in IMS 11 to support a Variable Trace capability that allows the calling component to write variable length trace data. This enhancement removes the restriction of predetermined fixed entry lengths.

When the trace table is externalized, the variable length entries are copied by the same asynchronous process as is used for trace tables using fixed length entries.

BPE Trace Enhancements …

- BPE Immediate External Trace facility
  - Addresses the data loss issue of the asynchronous copy process of the BPE External Trace facility
    - Immediately creates a copy of the trace entry in a dedicated copy buffer
  - Characteristics
    - Only supports variable length trace table entries
    - When the trace table is not externalized - EXTERNAL(NO):
      - The trace entry is created in the in-core trace table.
    - When the trace table is externalized - EXTERNAL(YES):
      - The trace entry is created in the in-core trace table and immediately copied to a dedicated copy buffer attached to the trace table.
      - The external trace asynchronous copy process ignores any trace table defined to use the Immediate External Trace facility.

114

The Immediate External Trace facility addresses the data loss issue of the asynchronous copy process of the BPE External Trace facility by immediately creating a copy of the trace entry in a dedicated copy buffer.

The Immediate External Trace facility works as follows:

- It is supported only for variable length trace table entries.

- When the trace table is not externalized, e.g., EXTERNAL(NO), then the trace entry is created in the in-core trace table.

- When the trace table is externalized, e.g., EXTERNAL(YES), then the trace entry is created in the in-core trace table and immediately copied to a dedicated copy buffer attached to the trace table.

The BPE external trace asynchronous copy process, introduced in IMS 10, ignores any trace table defined to use the new Immediate External Trace facility.

BPE Trace Enhancements …

- BPE Direct External Trace
  - Addresses the data loss issue of the asynchronous copy process
    - Creates the trace entry directly in a dedicated copy buffer without the overhead of copying the trace entry
    - Note: this is what the new IMS Connect recorder trace uses
  - Characteristics
    - Only supports variable length trace table entries.
    - When the trace table is not externalized - EXTERNAL(NO):
      - The trace entry is created in the in-core trace table.
    - When the trace table is externalized - EXTERNAL(YES):
      - The trace entry is created directly in the dedicated copy buffer attached to the trace table. No in-core entry is created.
      - The external trace asynchronous copy process ignores any trace table defined to use the Direct External Trace facility.

115

The Direct External Trace facility addresses the data loss issue of the asynchronous copy process without the overhead of copying the trace entry by creating the trace entry directly in the dedicated copy buffer.

The Direct External Trace facility works as follows:

- It is supported only for variable length trace table entries.

- When the trace table is not externalized, e.g., EXTERNAL(NO), then the trace entry is created in the in-core trace table.

- When the trace table is externalized, e.g., EXTERNAL(YES), then the trace entry is created directly in the dedicated copy buffer attached to the trace table. No in-core entry is created.

The BPE external trace asynchronous copy process ignores any trace table defined to use the Direct External Trace facility.

BPE Trace Enhancements …

- BPE Immediate External Write
  - Addresses the data loss issue of the asynchronous copy process and adds a layer of data integrity
    - Allows immediate write of buffered data to the external trace data set without waiting for the copy buffer to be filled
  - Characteristics
    - Only supports variable length trace table entries.
    - Only supports trace tables that use either the Immediate External Trace facility or the Direct External Trace facility.
    - As soon as the trace entry is created
      - Swaps in a free copy buffer to replace the current copy buffer
      - Passes the current copy buffer to the AWE processor to be written to the external trace data set

116

The Immediate External Write facility further addresses the data loss issue of the asynchronous copy process by allowing for the immediate write of buffered data.

The Immediate External Write facility works as follows:

- It is supported only for variable length trace table entries.

- It is further only supported for trace tables defined to use the Immediate External Trace facility or the Direct External Trace facility.

- As soon as the trace entry is created a free copy buffer is swapped in to replace the current copy buffer and the current copy buffer is passed to the AWE processor and written to the external trace data set.

BPE Trace Enhancements …

- New BPE trace table for NAME() parameter
  - **ERRV** – Error variable entry trace table
- Commands
  - BPE UPDATE TRACETABLE Command
    - UPDATE TRACETABLE NAME(**ERRV**) OWNER(BPE)…
    - UPD TRTAB NAME(**ERRV**) OWNER(BPE)…
  - BPE DISPLAY TRACETABLE Command
    - DISPLAY TRACETABLE NAME(**ERRV**) OWNER(BPE)…
    - Two new columns added to the display output

      #ENTRIES – # of trace entries that have been created in the trace table

      #CYCLES – # of times the trace table has been filled and wrapped back to the top

117

In support of the new trace functionality, a new trace table has been added to the BPE DISPLAY and UPDATE TRACETABLE commands. ERRV defines the new error variable entry trace table. Additionally, two new columns have been added to the DISPLAY TRACETABLE output to display the trace table entry and cycle counts.

Note that DISPLAY TRACETABLE command output displays the ERRV trace table when the following commands are used under any BPE address space:

DISPLAY TRACETABLE NAME(ERRV)
DISPLAY TRACETABLE NAME(E*)
DISPLAY TRACETABLE NAME(*)
DISPLAY TRACETABLE NAME(*) OWNER(BPE)

BPE Trace Enhancements …

- Commands…
  - BPE DISPLAY TRACETABLE Command example

Command:

F HWS1,DISPLAY TRACETABLE NAME(E*)

Output:

BPE0030I	TABLE	OWNER	LEVEL	#PAGES	EXT	#ENTRIES	#CYCLES
BPE0000I	ENVT	HWS	HIGH	200	NO	100019K	3907
BPE0000I	ERR	BPE	HIGH	2	NO	6	0
BPE0000I	ERRV	BPE	HIGH	8	NO	0	0
BPE0000I	ERRV	HWS	HIGH	8	YES	0	0

BPE0032I DIS TRTAB COMMAND COMPLETED

118

The example on this visual shows that the display includes the two new columns (#entries, #cycles) along with the new ERRV trace table entries.

BPE Trace Enhancements …

- Commands…
  - BPE DISPLAY TRACETABLE Command examples …

#Entries displayed in 1000's:

```
BPE0030I TABLE  OWNER  LEVEL  #PAGES  EXT  #ENTRIES    #CYCLES
BPE0000I ENVT   HWS    HIGH      200  NO     100019K      3907 ←
BPE0000I ERR    BPE    HIGH        2  NO           6         0
BPE0000I ERRV   BPE    HIGH        8  NO           0         0
BPE0000I ERRV   HWS    HIGH        8  YES          0         0
BPE0032I DIS TRTAB COMMAND COMPLETED
```

#Entries displayed in 1,000,000's:

```
BPE0030I TABLE  OWNER  LEVEL  #PAGES  EXT  #ENTRIES    #CYCLES
BPE0000I ENVT   HWS    HIGH      200  NO     100000M   3906250
BPE0000I ERR    BPE    HIGH        2  NO           6         0
BPE0000I ERRV   BPE    HIGH        8  NO           0         0
BPE0000I ERRV   HWS    HIGH        8  YES          0         0
BPE0032I DIS TRTAB COMMAND COMPLETED
```

#Cycles also has multiplier:

```
BPE0030I TABLE  OWNER  LEVEL  #PAGES  EXT  #ENTRIES    #CYCLES
BPE0000I ENVT   HWS    HIGH      200  NO    2560000M   100000K
BPE0000I ERR    BPE    HIGH        2  NO           6         0
BPE0000I ERRV   BPE    HIGH        8  NO           0         0
BPE0000I ERRV   HWS    HIGH        8  YES          0         0
BPE0032I DIS TRTAB COMMAND COMPLETED
```

Too much of a good thing:

```
BPE0030I TABLE  OWNER  LEVEL  #PAGES  EXT  #ENTRIES    #CYCLES
BPE0000I ENVT   HWS    HIGH      200  NO    OVERFLOW  OVERFLOW
BPE0000I ERR    BPE    HIGH        2  NO           6         0
BPE0000I ERRV   BPE    HIGH        8  NO           0         0
BPE0000I ERRV   HWS    HIGH        8  YES          0         0
BPE0032I DIS TRTAB COMMAND COMPLETED
```

119

Note that both the entries and cycles columns can have multipliers.

BPE Trace Enhancements …

- BPE Dump Formatter Support
  - Low-level formatting

```
----------------- IMS BPE LOW LEVEL DUMP FORMATTING OPTIONS Row 1 to 12 of 13

N <===== SPOOL OUTPUT? (Y OR N)            <====SYSOUT CLASS (Default Z)
     S = Select (default formatting)     Select choice(s) plus argument
     M = Select (minimum formatting)     values and hit enter to process
     X = Select (maximum formatting)     or UP/DOWN to scroll.
     T = Select (trace STCK formatted as time stamp)

Cmd  Option   Type      ARG           Argument description
v----------------------vvvvvvvv----------------------------------------------
    AWE      TYPE                     AWE blocks for specified AWE server type
    AWE      ADDR                     AWE blocks for specified AQHE address
    BUFP     TYPE                     Buffer pool blocks for specified pool
    BUFP     ADDR                     Buffer pool blocks for specified BPDB addr
    CBTE     TYPE                     CBTE blocks for specified block type
    CBTE     ADDR                     CBTE blocks for specified CBTE address
    DISP     TYPE                     Dispatcher blocks for specified TCB type
    DISP     ADDR                     Dispatcher blocks for specified TCB addr
    THREAD   TYPE                     Thread blocks for specified thread type
    THREAD   ADDR                     Thread blocks for specified thread ECB adr
    TRACE    TYPE       ERRV          Trace formatting of specified trace table
    USRX     TYPE                     User exit blocks for specified exit type

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

120

This visual and the next several show the BPE Dump Formatter support for the new ERRV trace table.

BPE Trace Enhancements …

- Formatting and printing external trace records with a batch job

```
//BPEEXTPR JOB ...
//***************************************************************/
//* Job to print all traces in a BPE external trace data set. */
//***************************************************************/
//JOBLIB   DD DSN=IMS.SDFSRESL,DISP=SHR
//IPCSDMP  EXEC PGM=IKJEFT01,REGION=8M
//SYSTSPRT DD SYSOUT=*
//IPCSPRNT DD SYSOUT=*
//INDEX    DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//IPCSPARM DD DSN=SYS1.PARMLIB,DISP=SHR
//SYSTSIN  DD *
  DELETE 'SYS1.IPCSDDIR'
  ALLOC SP(1) TRACK VOL(333333)
  DEFINE CLUSTER (NAME('SYS1.IPCSDDIR') VOLUMES(333333)) +
    INDEX(NAME('SYS1.IPCSDDIR.DDX') TRACKS(1 1)) +
    DATA( NAME('SYS1.IPCSDDIR.DDD') +
    CYLINDERS(1 1) BUFSP(X'10000') KEYS(128 0) CISZ(X'1000'))
  IPCSDDIR 'SYS1.IPCSDDIR'
  ALLOC FILE(IPCSDDIR) DA('SYS1.IPCSDDIR') REUSE SHR
  ALLOC FILE(INFILE)   DA('BPEEXTRC.GDG01.G0001V00') REUSE SHR     ← External Trace data set name
  IPCS NOPARM SETDEF  DSN('BPEEXTRC.GDG01.G0001V00') +             ← External Trace data set name
    NOPROBLEM PRINT NOTERMINAL +
    VERBX BPETRFM0 'TRACE(TYPE(ALL))'                               ← VERBX command and parameters
  END
  DELETE 'SYS1.IPCSDDIR'
/*
```

121

The external trace records can also be printed through a batch job process.

BPE Trace Enhancements …

- BPETRFM0 parameter syntax

```
                                      ┌─,────────────────┐
                                      ▼                  │
►►─COMP(─┬─ALL────────────┬─)─TRACE(─TYPE(─┬─ALL──────────────┴─))──────────►
         ├─BPE────────────┤                └─trace table name─┘
         └─component name─┘

►─┬──────────────────────────────────────┬─┬──────────────────────────────────────┬─►
  └─SDATE(yyyyddd)─┬───────────────────┬─┘ └─EDATE(yyyyddd)─┬───────────────────┬─┘
                   └─STIME(hhmmss)─────┘                    └─ETIME(hhmmss)─────┘

►─┬──────────────────┬─┬───────────────┬─────────────────────────────────────────►◄
  └─CSTCK(─┬─Y─┬─)───┘ └─UL(─┬─U─┬─)───┘
           └─N─┘             └─L─┘
```


The instructions for formatting a BPE trace entry with a batch job, documented in the IMS Diagnosis Guide, are not changed by this enhancement, but documentation on the VERBX BPETRFM0 parameter string is now included.

- COMP() - specify ‘ALL’ to format all trace records, ‘BPE’ to format BPE trace records, or a specific component name, e.g., CQS, OM, RM, SCI, HWS – to format trace records only for that component.
- TYPE() - specify ‘ALL’ to format trace records for all trace tables or specify a specific trace table name to format records only for that trace table type.
- SDATE() - specify a starting date for the trace entries in Julian format (yyyyddd). Trace entries with an STCK value prior to the specified date are filtered and not formatted.
- STIME() - specify a starting time for the trace entries in 24 hour format (hhmmss). Trace entries with an STCK value prior the specified time are filtered and not formatted. SDATE() is required with STIME().
- EDATE() - specify an ending date for the trace entries in Julian format (yyyyddd). Trace entries with an STCK value after the specified date are filtered and not formatted.
- ETIME() - specify an ending time for the trace entries in 24 hour format (hhmmss). Trace entries with an STCK value after the specified time are filtered and not formatted. EDATE() is required with ETIME().
- CSTCK() - specify ‘Y’ to have the value for each trace entry printed in JDAYTIME format (DDD HHMMSS.thmiju).
- UL() - specify ‘L’ if the specified filtering time is based on the local time in the trace record or ‘U’ if the specified filtering time is based on UTC.

BPE Trace Enhancements …

- VERBX command BPETRFM0 parameters…

```
IPCS +
  NOPARM +
  SETDEF DSN('BPEEXTRC.GDG01.G0001V00') +
  NOPROBLEM PRINT NOTERMINAL +
  VERBX BPETRFM0 +
   'COMP(HWS) +
    TRACE(TYPE(RCTR)) +
    SDATE(2008080) STIME(110909) +
    EDATE(2008090) ETIME(140000) +
    UL(L) +
    CSTCK(Y)'
END
```

The example VERBX BPETRFM0 parameter will print:

- HWS trace entries
- In the RCTR trace table
- If the entry was created after 2008 080 11:09:09 local time
- And before 2008 090 14:00:00 local time
- With STCK values printed in JDAYTIME format (DDD HHMMSS.thmiju)

This visual provides a more specific example of a VERBX BPETRFM0 parameter set of specifications.

IMS Connect Recorder Trace Enhancement

- The second part of the solution:

- A new IMS Connect-owned BPE trace table name RCTR
  - Uses the BPE Direct External Trace facility

- With enhanced code in the Recorder Trace process
  - Re-routes trace data to the RCTR table (BPE External Trace dataset) if the following conditions are met:
    - The old recording facility using the HWSRCORD DD is not active
    - The RCTR trace table has been activated and EXTERNAL(YES) specified
    - The BPE External Trace dataset facilities have been installed

124

A new IMS Connect-owned BPE trace table named RCTR is defined with the BPE Direct External Trace option enabled. Enhanced code in the Recorder Trace record process re-route trace data to the RCTR table, and thereby to the BPE External Trace data set. This capability replaces the older recording facility that uses the HWSRCORD data set.

IMS Version 11

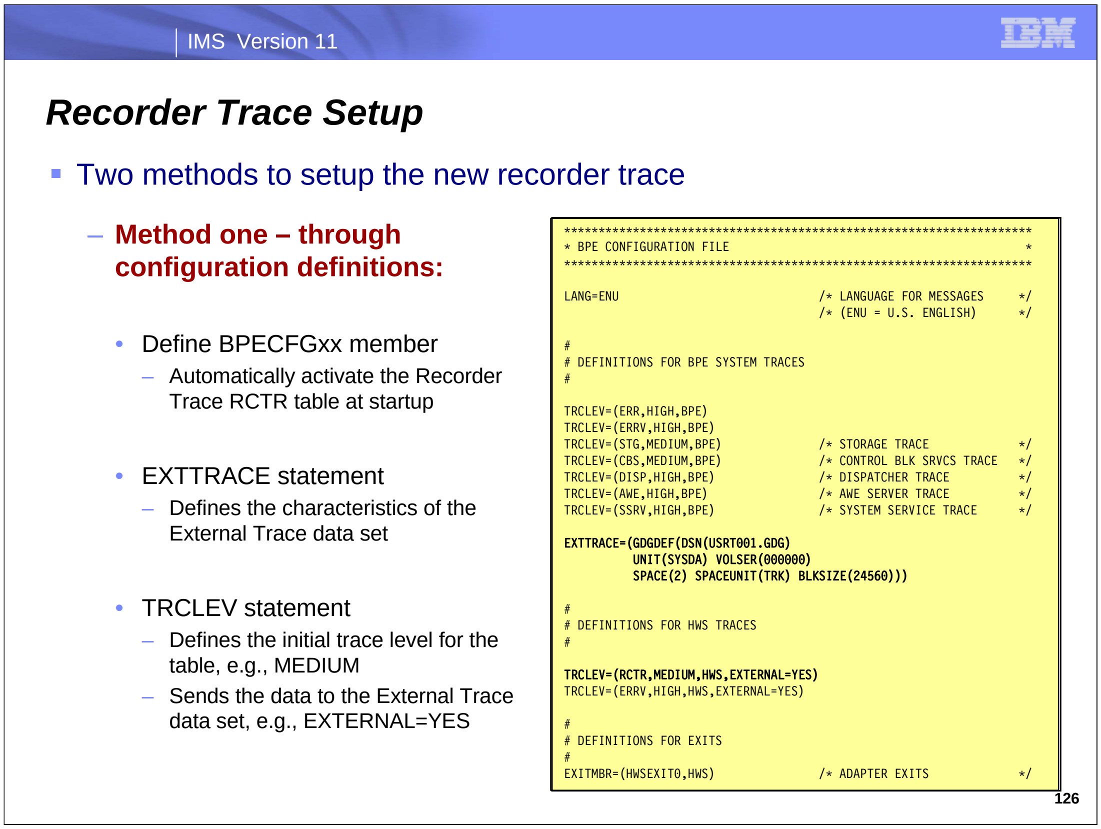
IMS Connect Recorder Trace Enhancement …

- Compatibility
  - For downward compatibility, the old Recorder Trace process continues to be supported for one more version
  - The old and new facilities cannot be used concurrently, but can be used serially
    - Allows switching from old to new and back
      - Without stopping the IMS Connect address space
  - The old Recorder Trace data set takes precedence
    - If the RCTR trace table is enabled (in-core or external) and the old Recorder Trace process is started
      - Recorder Trace data is directed to the recorder data set
      - When the recorder data set fills or the old process is manually stopped
        - Recorder Trace data reverts back to the RCTR table

125

For downward compatibility purposes and to allow users time to migrate to the new facility, the old Recorder Trace process continues to be supported for one more version. Note, however, that the old and new facilities cannot be used concurrently. They can be used serially, switching from old to new and back, without the need to stop the IMS Connect address space.

If the old recorder trace process is started while the BPE capability is running, the old recorder trace data set takes precedence and the recorder trace data is directed to the data set defined in the HWSRCORD DD statement. If this data set fills up or the old process is manually stopped then the tracing reverts to the RCTR table.



The new Recorder Trace facility can be started using one of two methods:

Method one is to include the BPECFGx TRCLEV(RCTR,MEDIUM,HWS,EXT=YES) parameter statement in the appropriate BPE Configuration Parameter PROCLIB member and ensure that the EXTTRACE statement is also defined to include the external trace data set characteristics.

Note the EXTERNAL(YES) option on the BPE UPDATE TRACETABLE command must be defined if the Recorder Trace data is to be written to the BPE External Trace data set. If the BPE UPDATE TRACETABLE command is issued without the EXTERNAL option, which defaults to EXTERNAL(NO), or if EXTERNAL(NO) is specified, the Recorder Trace data will only be written to the in-core BPE trace tables. Also note that all Recorder Trace trace entries are generated at the BPE UPDATE TRACETABLE level of MEDIUM. Only MEDIUM and NONE have meaning for this initial implementation of the RCTR table.

Recorder Trace Setup …

- Two methods to setup the new recorder trace …
  - **Method two – through commands (WTOR or z/OS Modify) :**
    - To use the old Recorder Trace data set issue a command to start the recorder:
      - R ##,RECORDER OPEN
      - F HWS1,UPDATE MEMBER TYPE(IMSCON) START(TRACE)
    - To use the new Recorder Trace RCTR table issue a command to stop the old recording facility:
      - R ##,RECORDER CLOSE
      - F HWS1,UPDATE MEMBER TYPE(IMSCON) STOP(TRACE)
    - And issue a BPE UPDATE TRACETABLE command to start the RCTR trace table:
      - To create Recorder Trace entries in the BPE in-core table:
        - F HWS1, UPD TRTAB NAME(RCTR) OWNER(HWS) LEVEL(MEDIUM)
      - To write Recorder Trace entries to the External Trace data set:
        - F HWS1, UPD TRTAB NAME(RCTR) OWNER(HWS) LEVEL(MEDIUM) EXTERNAL(YES)

127

The second method of setting up the new recorder trace is through commands.

The first set of commands on the visual show how to start the old recorder tracing facility by issuing a response to the outstanding WTOR or by issuing a zOS modify (F) command UPDATE request.

To use the new tracing capability, the old recorder trace must first be stopped if it is active. Once the old recorder is no longer active, the BPE UPDATE TRACETABLE command can be issued to start the new facility. If the UPDATE TRACETABLE command does not specify EXTERNAL(YES), even if it specifies a level other than NONE, no trace data will be written to the BPE External Trace data set, only to the in-core tables. Note that while the UPDATE TRACETABLE accepts all documented values for the LEVEL() parameter, including LOW, MEDIUM, and HIGH, only MEDIUM and NONE have meaning for the initial implementation of the RCTR table.

Recorder Trace Enhancements - Commands

- More information on the BPE commands
  - New HWS trace table for NAME() parameter
    - **ERRV** – Error variable entry trace table
    - **RCTR** – Recorder Trace trace table
  - F *addresspace,* DISPLAY TRACETABLE Command

    DISPLAY TRACETABLE NAME(**ERRV**) OWNER(HWS)…
    DISPLAY TRACETABLE NAME(**RCTR**) OWNER(HWS)…
  - F *addresspace*, UPDATE TRACETABLE

    UPDATE TRACETABLE NAME(**ERRV**) OWNER(HWS)…
    UPDATE TRACETABLE NAME(**RCTR**) OWNER(HWS)…
    UPD TRTAB NAME(**ERRV**) OWNER(HWS)…
    UPD TRTAB NAME(**RCTR**) OWNER(HWS)…

128

Both the DISPLAY and UPDATE TRACETABLE commands have the following trace table definitions added to the IMS Connect (HWS) list:

RCTR	IMS Connect Recorder Trace table
ERRV	IMS Connect error variable entry trace table

Recorder Trace Enhancements – Commands…

- Example

```
F HWS1,DISPLAY TRTAB NAME(*) OWNER(HWS)

BPE0030I TABLE  OWNER  LEVEL  #PAGES  EXT  #ENTRIES   #CYCLES
BPE0000I ADPT   HWS    HIGH       20  NO          0         0
BPE0000I CMDT   HWS    HIGH       20  NO         24         0
BPE0000I ENVT   HWS    HIGH      200  NO         18         0
BPE0000I ERRV   HWS    HIGH        8  YES         0         0
BPE0000I HWSI   HWS    HIGH      200  NO         63         0
BPE0000I HWSN   HWS    HIGH       20  NO          3         0
BPE0000I HWSO   HWS    HIGH       20  NO          6         0
BPE0000I HWSW   HWS    HIGH      200  NO         91         0
BPE0000I LEPS   HWS    HIGH      200  NO          0         0
BPE0000I ODBM   HWS    HIGH       20  NO          2         0
BPE0000I ODDR   HWS    HIGH       20  NO          0         0
BPE0000I OMDR   HWS    HIGH       20  NO          2         0
BPE0000I OSDR   HWS    HIGH       20  NO          0         0
BPE0000I OSOC   HWS    HIGH       20  NO          0         0
BPE0000I OTMA   HWS    HIGH      200  NO        323         0
BPE0000I PCDR   HWS    HIGH       20  NO          5         0
BPE0000I PCIF   HWS    HIGH       20  NO          0         0
BPE0000I RCTR   HWS    MEDIUM     20  YES         4         0
BPE0000I RRSI   HWS    HIGH       20  NO        108         0
BPE0000I TCPI   HWS    HIGH      200  NO         48         0
BPE0032I DISPLAY TRTAB COMMAND COMPLETED
```

129

If the tracetable name is set to *, then all trace table entries associated with the address space are listed.

Recorder Trace Record

- And when the records are printed
  - This is what the printed recorder trace record entry looks like

```
RCTR trace table entry:
         Code:     00                  Record #: 9
         Subcode:  8B                  Continue: 0
         TimeStmp: 096 180948.574553
 TTVE: 00000000

         Variable entry prefix:
         LL....... 05C0                REC#..... 00000009
         ZZ....... 05C0                CONT..... 00000000
         VLEN..... 000005A0
 TTE: 00000010
         Variable entry fixed section:
         CODE..... 00
         SCDE..... 8B
         B1B2..... 0000
         WD01..... 00008608
         STCK..... C233A4AA  ED359CC2
         Data +00: |......f.B.u....B|
 Data: 00000020  Length: 1440
Variable entry variable section:
 Offset  0        4        8        C         0        4        8        C          EBCDIC Data
 ------  -----------------------------------  -----------------------------------  ---------------------------------
+000000  00000000 C9C3D6D5 D9C30052 00000877  18094857 0108096F 00000000 00000000  |....ICONRC.............?........|
+000020  E3D4C1D7 D7D3F3F1 C233A4AA EC40D788  C233A4AA EC69C288 00000000 00000000  |TMAPPL31B.u.. PhB.u...Bh........|
+000040  00000000 00000000 00000000 00000000  00000000 00000000 00000000 5CC9D7C2  |............................*IPB|
+000060  00000072 00500000 5CE2C1D4 D7D3C55C  00000000 00630000 E3D4C1D7 D7D3F3F1  |.....&..*SAMPLE*........TMAPPL31|
+000080  00200140 C1D7D6D3 F1F14040 C9D4E2F1  40404040 40404040 40404040 E4E2C5D9  |... APOL11  IMS1            USER|
+0000A0  F0F0F0F1 40404040 40404040 5C5C5C5C  5C5C5C5C 001A0000 C1D7D6D3 F1F140D9  |0001        ********....APOL11 R|
+0000C0  C5E2D740 E3D9C1C3 C54DC4D3 C95D0004  00000000 00000000 00000000 00000000  |ESP TRACE(DLI)..................|
+0000E0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  |................................|
```


When printed, the trace records provide a readable view of the trace data.

Recorder Trace - Operational Considerations

- Automation
  - Mechanisms that start, stop, or monitor the old Recorder Trace facility should be deleted once the new facility is implemented
  - If not already set up to manage the BPE external dataset facility, it should be considered.
  - Existing automation that examines BPE DISPLAY TRACETABLE or UPDATE TRACETABLE command output may need to be updated to account for the new information
- Automatic setup of the new Recorder Trace facility at IMS Connect startup time
  - Define the RCTR TRCLEV statement in the BPE configuration definition
- Remove the HWSRCORD DD statement to disable the old recorder function

131

From an operations perspective:

- If automation related to the old Recorder Trace facility is in place, it will need to be deleted once the new facility is implemented.

- If automation already exists for the BPE External Trace data set, no further effort is required. On the other hand, if automation has not already been implement to manage the data sets created by the BPE External Trace facility, it should be considered.

- Automation that examines BPE DISPLAY TRACETABLE or UPDATE TRACETABLE command output may need to be updated to account for the new information.

Automatic startup of the new Recorder Trace facility at IMS Connect startup time should be done by including the TRCLEV statement with the RCTR specification in the appropriate BPE configuration parameter PROCLIB member. To remove the old facility, the HWSRCORD DD must be deleted from the IMS Connect startup procedure

Recorder Trace Benefits

- Benefits
  - Improves IMS Connect product reliability by reducing system outages caused by abends in the Recorder Trace process
  - Eliminates Recorder Trace dataset full conditions and the resulting loss of diagnostic data
  - Improves the reliability of diagnostic information to shorten or streamline the problem determination process
  - Allows the user more flexibility and control over the amount of diagnostic data recorded

Other Capabilities

- Support for Callout and Open DB are discussed in the Integration section of the class

IMS Connect

Migration Considerations

IMS Connect Migration Considerations

- HWSCFGx
  - Single SSLPORT restriction
    - Ensure that the SSLPORT parameter only specified one port
    - Previous releases allow initialization to complete with multiple SSL ports
      - IMS 11 produces U3401 abend at initialization
  - IMS Connect display output has new information and fields that reflect the enhancements discussed in this section
    - Modify automation programs and MTO documentation as needed
- New/changed messages that affect automation and MTO documentation
  - HWSP1410W enhancement for errors occurring during storage release
  - HWSX0908W - issued if HWSIMSO0/HWSIMSO1 continue to be specified
  - HWSS0772W and HWSS0773I – issued in support of WARNSOC/WARNINC

135

The next two visuals document the migration considerations for IMS Connect.

IMS Connect environments, in previous releases, 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. In IMS 11, IMS Connect initialization will fail if multiple SSL ports are specified. The HWSCFGx member needs to be modified to only specify one port.

Several new specifications in the HWS, TCPIP and DATASTORE statements of the HWSCFGx configuration file are added to the display output of commands such as VIEWHWS, VIEWPORT, VIEWDS, etc. Automation programs and MTO documentation should be modified to recognize these new fields.

Automation programs that read the output of IMS Connect displays or query the HWSP1410W message need to be aware of the new information and fields that have been added by the IMS 11 enhancements. Similarly, Master Terminal Operators (MTOs) that issue IMS Connect commands should understand that additional information is provided.

Message HWSX0908W is issued if the old exits HWSIMSO0/HWSIMSO1 continue to be specified in the IMS Connect configuration member.

If WARNSOC and WARNINC are specified in the TCPIP HWSCFGx statement then new messages will be issued when the warning level is reached (HWSS0772W) and when the number of sockets falls below the warning level (HWS0773I).

IMS Connect Migration Considerations

- Required re-assembly of IMS Connect user message exits
  - Incorporates the expansion of the HWSEXPRM macro and XIBDS
- TCP/IP Automatic Reconnect
  - Remove directions to issue OPENPORT command from MTO documentation or automation for situations when the network bounces while IMS Connect is active
- Generated Client ID
  - Replace the resource adapter with the new one to take advantage of the support
- Recorder Trace
  - Old tracing remains in effect until the HWSRCORD DD is removed
    - BPE external trace datasets may contain new variable length trace entries as well as fixed length trace entry data

136

Because the HWSEXPRM macro has been expanded, IMS Connect exit routines that invoke the macro must be re-assembled. As a reminder, from the OTMA Resource Monitoring section, the XIBDS (Exit Interface Block Data Store Entry) has also been expanded.

Operator commands that are issued to ensure that IMS Connect reestablishes connectivity with a TCP/IP network are no longer needed. 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.

To take advantage of the Generated Client ID function, the IMS TM resource adapter must be replaced with the new version.

The new Recorder 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. One the new function has been enabled, new RCTR entries in the BPE external trace datasets will be introduced as variable length trace entries.

IMS Connect - Summary and Overall Benefits

- System reliability, manageability and performance
  - Automatic re-establishment of connectivity to TCP/IP after a network failure
  - Configuration level overrides to control environmental conditions such as message timeouts, TCP/IP stack settings, error conditions, etc.
  - Enhanced diagnostic information including the addition of warning messages
  - Improved hashing technique for client ids
- Usability
  - Support for the modification of input messages from TCP/IP before they are submitted to IMS Connect
- Serviceability
  - Alternative recorder trace capability to eliminate the previous issues with dataset full conditions and the resulting loss of diagnostic data

IMS Version 11

Integration

Information Management software

Integration

- IMS Web 2.0 Solution for InfoSphere MashupHub
- IMS TM Resource Adapter 10.2
- IMS 10 Synchronous Callout
- IMS 10 Database Web Services
- IMS 10 Soap Gateway
- IMS 11 Open Database

IMS Web 2.0 Solution for InfoSphere MashupHub

IMS Web 2.0

- An IMS capability that provides WEB 2.0 access to IMS transactions

WEB 2.0

IMS™ Info 2.0 is the IMS Web 2.0 solution.

WEB 2.0

- ???
- A second-generation of web communities and hosted services
  - Supports a transition from websites containing isolated information to interlinked environments
    - The idea is that the web itself is a computing platform
  - Leverages one main approach to web APIs:
    - REST (Representational State Transfer)
      - Uses a unique URL to represent an object
      - Uses an HTTP GET verb to access the object and POST, PUT, or DELETE to modify the object
    - SOAP (Simple Object Access Protocol)

5

To understand the IMS Web 2.0 support, some terminology needs to be explained.

Web 2.0 describes an evolving trend in the use of web technology and design with a goal of enhancing information sharing and collaboration. Since its introduction, Web 2.0 concepts have led to the development and evolution of web-based communities and hosted services such as social-networking sites, video sharing site, wikis, blogs, etc. Although the term suggests a new version of the web, it does not refer to an update to any technical specifications, but to changes in the ways software developers and end-users view the web. The idea is that the web or internet is itself a computing platform rather than a disparate set of connection environments.

Web 2.0 leverages two approaches to the web APIs, REST (representational state transfer) and SOAP (simple object access protocol).

Web 2.0 --- XML, RSS and ATOM Feeds

- Addresses the continued goal of simplification
  - Feeds are XML documents containing lists of related information composed of a number of items, known as "entries", each with an extensible set of attached metadata (http://www.ietf.org/rfc/rfc4287.txt)
    - Web feeds allow programs to check for updates published on a web site

***Atom** is an XML-based Web content and metadata syndication format, and an application-level protocol for publishing and editing Web resources (AtomEnabled.ORG)*

***RSS** is a web feed formats used to publish frequently updated works, e.g. blog entries, news headlines, audio, and video – in a standardized format*

- Feeds and REST
  - POST creates XML, RSS, Atom entries and media files
  - GET retrieves the entry point Introspection document, collections (represented as feeds), and individual resources (entries or media files)
  - PUT updates entries and media files
  - DELETE removes an entry or a media file

6

Feeds are XML documents containing lists of related information composed of a number of items, known as "entries", each with an extensible set of attached metadata. They are intended to support synchronization between publishers and consumers.

The name Atom applies to a pair of related standards. The Atom Syndication Format is an XML language used for web feeds, while the Atom Publishing Protocol (AtomPub or APP) is a simple HTTP-based protocol for creating and updating web resources. Web feeds allow software programs to check for updates published on a web site. A feed contains entries, which may be headlines, full-text articles, excerpts, summaries, and/or links to content on a web site, along with various metadata.

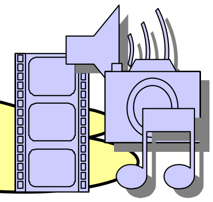
RSS is a family of web feed formats used to publish frequently updated works – such as blog entries, news headlines, audio, and video – in a standardized format. An RSS document includes full or summarized text, plus metadata such as publishing dates and authorship. RSS refers to the following formats: "Really Simple Syndication (RSS 2.0)", “RDF Site Summary (RSS 1.0 and RSS 0.90)", or "Rich Site Summary (RSS 0.91)".

To provide a web feed, a site owner may use specialized software (such as a content management system) that publishes a list (or "feed") of recent articles or content in a standardized, machine-readable format. The feed can then be downloaded by web sites that syndicate content from the feed, or by feed reader programs that allow Internet users to subscribe to feeds and view their content

IMS Version 11

The Goal --- > Mashups

Comes from music terminology:
music that contains songs already released by other artists



- For the web:
  - An application that combines data or content **(FEEDs)** from more than one source into a single integrated view
- Three types
  - Consumer
    - Integrates data elements from multiple sources with a simplified front-end
      - E.g., Google Maps
  - Data and Enterprise
    - Mixes data of similar types from different sources with a graphical front-end
      - E.g., Manufacturer's product description map on closest retailers to a zip code
  - Business
    - combination of all the above, focusing on both data aggregation and presentation, with an added bonus of collaborative functionality
      - E.g., telecommunications service where service elements come from more than one source such as the base service from company A, a ringback tone from company B, a voicemail service from company C, etc.

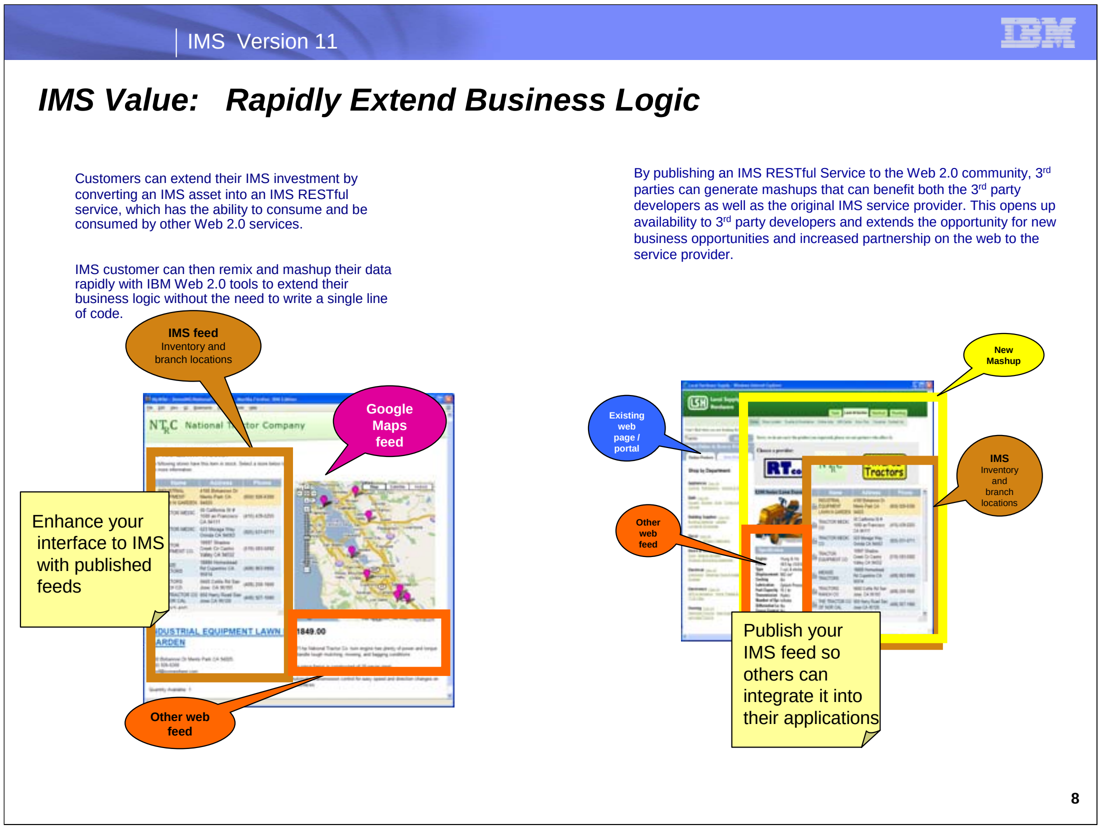
7

Additionally, the term Mashup is an important part of the Web 2.0 environment and describes a web application that combines data from more than one source into a single integrated tool. There are three categories of mashups: consumer, data and enterprise, and business.

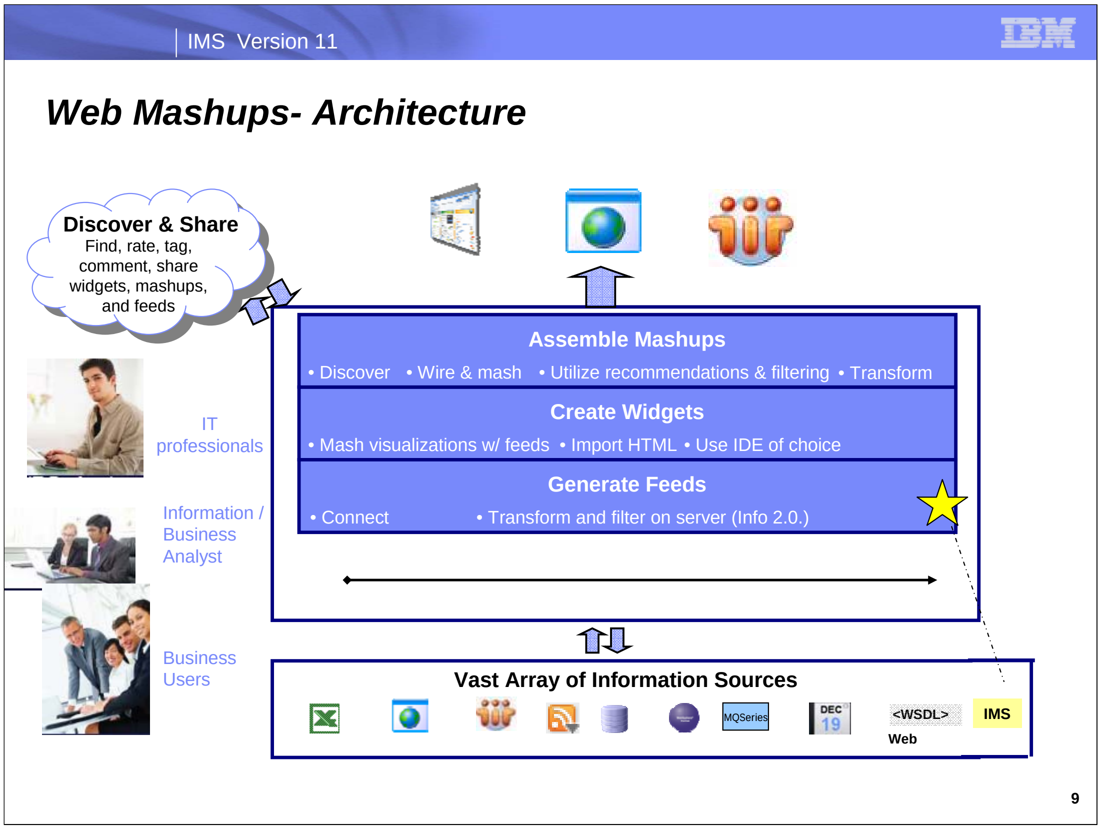
A good example of a simple consumer mashup is the use of cartographic data from Google Maps to add location information to real-estate data, thereby creating a new and distinct web service that was not originally provided by either source.

A second kind of mashup mixes data of similar types from different sources. For example a manufacturer's web site can produce information about a specific product and dynamically provide a list and map of the closest retailers to the end user.

A business mashup would combine the other two types but could integrate information into a single view from a variety of collaborating servers and enterprises.



So what does this all have to do with IMS? The next two visuals provide examples of extending IMS transactions as web feeds.



In "Understanding Mashup Architecture," Thomas Howe describes three layers:

The top layer or the mashup layer aggregates and presents information supplied in the layers below it. The aggregation is normally done through applications that take advantage of mashup tools to provide graphical presentations and scripting tools to present information as web pages. IBM's QEDWiki is an example of a collaborative tool that provides document centric views.

The middle layer provides governance, business rules and workflow orchestration for the enterprise. Essentially, this layer makes sure that resources are used by appropriate people in an appropriate manner, and runs business processes for the company such as order management, supply notifications and risk triggers.

The lowest layer consists of the many data and functional services available to the enterprise. For instance, for a wall street firm, one data service may be a constant flow of orders made on the stock exchanges, or it may be an on demand service that calculates the current risk profile of a fund or position. For health care companies, it may be a database of all of the procedures performed on a patient by a department. Sometimes these services are actually delivered over the Internet and are provided by a variety of vendors. Mashups allow enterprises to share there internal services to other entities outside the firewall.

How do you do it with IMS?

- IBM Mashup Center includes two components:
  - **InfoSphere MashupHub** for creating, storing, transforming and remixing feeds, and for sharing with the Web 2.0 community.
  - **Lotus® Mashups** for rapidly assembling a variety of personal, enterprise, and Web content onto a Web page or into an application.
- IMS Web 2.0 is a plugin to the InfoSphere MashupHub V1.0
  - Includes the ability to:
    - Transform IMS assets into RESTful services (XML, ATOM, or RSS feeds) that can consume or be consumed by other Web 2.0 services
    - Compose "widgets" / HTML segments into a composite User Interface (UI) or an application template
    - Bind services (data or function) to the composite UI

10

The IMS Web 2.0 solution is embedded in the InfoSphere™ MashupHub and bundled in the IBM® Mashup Center. The IBM Mashup Center provides a Web interface that can unlock information buried in information silos. The IBM Mashup Center includes two components: InfoSphere MashupHub for creating, storing, transforming and remixing feeds, and for sharing with the Web 2.0 community; and Lotus® Mashups for rapidly assembling a variety of personal, enterprise, and Web content onto a Web page or into an application.

IMS Web 2.0 allows you to:

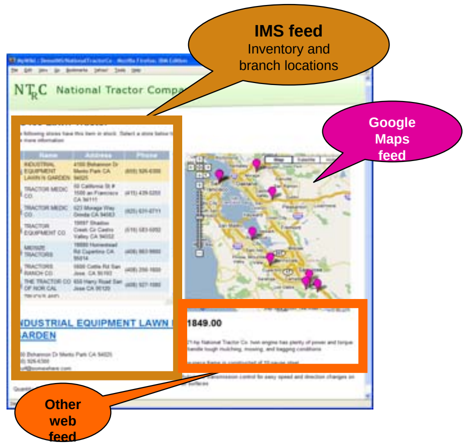
Create IMS feeds from IMS transactions that run on IMS with integrated IMS Connect. Both COBOL and PL/I applications are supported. An IMS feed is the connection information to the host IMS system using an XML-based file format (Atom or RSS).

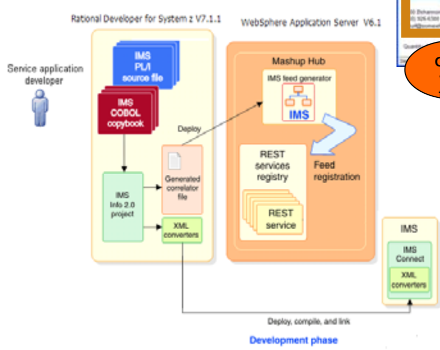
Easily customize IMS transactions without the need to modify the original application. Through a Web interface, you can specify which input parameters to expose to users, as well as the default parameter values to invoke the feed. Feeds can be further restructured and customized by using the operators and functions in InfoSphere MashupHub.

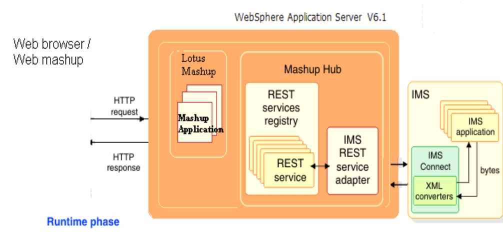
Utilize the tooling support in Rational Developer for System z Version 7.1.1 to generate the required XML converter driver and correlator. The XML converter driver is used by IMS Connect to transform the data between XML and bytes. The correlator is used by IMS Info 2.0 to map the request and response messages to the input and output data structures of the IMS application. IMS Version 10 customers are entitled to two complimentary, unwarranted copies of Rational Developer for System z Version 7.1.

IMS Web 2.0

Extends the IMS investment by converting an IMS asset into an IMS RESTful service, which has the ability to consume and be consumed by other Web 2.0 services.







11

IBM Mashup Center Features

- **A visual tool for creating, storing, transforming, and remixing feeds to be utilized in mashups, and a central catalog for users to tag, rate, and share mashable assets**

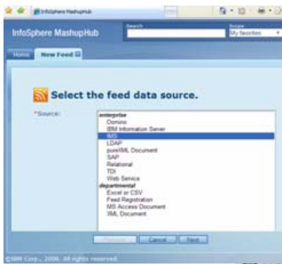
Transform and Mix

Import feeds Merge feeds

Filter feeds Publish feeds

Annotate feeds Transform feeds

Group / Sort / Union feeds



Create feeds

Domino LDAP

pureXML SAP TDI

Web Service

IBM Information Server

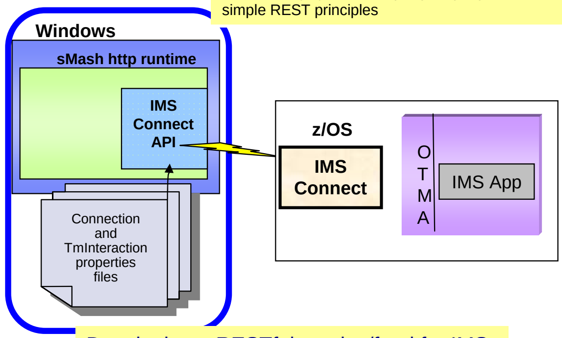
IMS Transaction Excel or CSV MS-Access XML Document

Enterprise Databases (IMS, DB2, etc) Feed Registration

WebSphere sMash and IMS Web 2.0

- **WebSphere sMash** is a development and execution platform for quickly building agile, web-based applications
  - Supports some of today's hottest dynamic scripting languages – PHP (**P**HP: **H**ypertext **P**reprocessor) and Groovy– and provides an agile web application development environment
  - JVM-based runtime (Not JEE)
  - Allows you to unleash and reuse enterprise content, including SOA services, as RESTful services
  - Is based on the highly-acclaimed public incubator and developer community, Project Zero
  - The IBM Reliable Transport Extension for WebSphere sMash v1.1 provides the capability for apps to interact using an asynchronous, reliable queue based transport.
  - WebSphere sMash Developers Edition v1.1 is available for free download and limited deployment. It contains browser-based tooling and an Eclipse plug-in.
  - WebSphere sMash licenses are purchased when deploying it into production.

WebSphere sMash enables developers to build applications in the Web 2.0 style by easily composing and pulling together preexisting assets using dynamic scripting languages and simple REST principles



Developing a RESTful service/feed for IMS Transactions
- Add the IMS Connect API for Java (a Jar file) to the sMash Application classpath
- Code connection and interaction information or load from a properties file.

Considerations

- Messages
  - IRW0xxx
- Restrictions
  - Commit Mode 1 and Sync Level None support only
  - Conversational transactions are not supported
  - Only COBOL copybook and PL/I source files are supported
  - Multi-segment messages are not supported.
  - Callout requests from IMS applications are not supported
- Environments supported
  - Windows 2003 Server
  - Internet Explorer V6, Internet Explorer V7, Firefox V2, and Safari V3

14

Messages produced by the IMS Web 2.0 support all beginning with IRW0xxx.

When planning the use of this capability, several restrictions need to be understood:

Interaction through IMS Connect is limited to Commit Mode 1 and Sync Level None. There is no support for confirmation or syncpoint processing.

Only non-conversational transactions can be invoked. Conversational transactions are not supported

Only COBOL copybook and PL/I source files are supported. MFS source cannot be used.

Only single-segments can be accessed. Multi-segment messages are not supported

Callout requests from IMS applications are not supported.

Software Requirements

- Software requirements
  - IMS 10 / IMS Connect 10 or later
  - Rational Developer for System z V7.1.1+
  - InfoSphere Mashup Center V1.0 which includes
    - InfoSphere MashupHub Enterprise Edition
    - Lotus Mashups
    - WebSphere Application Server V6.1.0.13
- Benefits
  - Positions IMS as a participant in the Web 2.0 community
    - Provides access to IMS resources as RSS, ATOM, or XML feeds

Currently the InfoSphere Mashup Hub Enterprise Edition installer packages both WebSphere and Apache Derby (open source relational database implemented in Java). There is no option to install on a preexisting instance of WebSphere.

Future releases will support installation on preexisting WebSphere instances and support for the Linux Operating System.

IMS Web 2.0 enables the integration of existing IMS assets into Web 2.0 mashup and application solutions. Through this capability access to IMS transactions can now also be accomplished through RSS, ATOM, or XML feeds.

How do you do it with IMS databases?

- IBM Mashup Center V2.0
  - Provides the ability to:
    - Aggregate, manipulate, and restructure data or content feeds
  - Includes :
    - The SQL Query Builder for creating feeds
  - Requires IMS Enterprise Suite DLIModel Utility
    - Supplies IMS Database Metadata

16

A *data mashup* is a feed that you create by taking a source feed (IMS DB data) and applying operators and functions to filter and restructure source data.

For an IMS database feed, The IBM Mashup Center V2.0 communicates with IMS through the IMS Universal DB resource adapter, a software component in IMS Version 11 that provides SQL-based database connectivity to access IMS databases over TCP/IP.

Because access to IMS databases is through SQL queries, an IMS database feed is created by selecting Enterprise Database (JDBC) as the feed source, in the same way as the other enterprise databases that are supported by IBM Mashup Center.

IMS 11 DB Web 2.0

- Capabilities
  - Create feeds for IMS data
  - DLIModel generates the metadata for feed input
  - IBM Mashup Center 2.0 for Enterprise Data Feed generation
  - IBM WebSphere Application Server runtime
  - IMS 11 Universal DB resource adapter communicates with IMS Connect
  - IMS 11 Connect communicates with IMS Open Database Manager
  - IMS 11 ODBM sends/returns IMS data

17

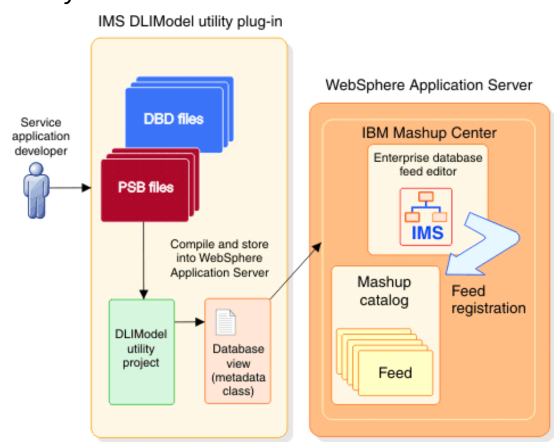
For feeds that access IMS databases, the IMS Universal DB resource adapter is used to communicate between IMS Connect and IBM WebSphere® Application Server, which is shipped with IBM Mashup Center. The IMS Universal DB resource adapter translates SQL queries from IBM Mashup Center and communicates with the Common Service Layer (CSL) Open Database Manager (ODBM) through IMS Connect by using the open standard Distributed Relational Database Architecture™ (DRDA®) as the low-level communication protocol.

WebSphere Application Server is installed as part of the IBM Mashup Center installation process.

Developing an IMS 11 database Feed

- Develop an IMS DB feed using:
  – IMS Enterprise Suite DLIModel utility
  – IBM Mashup Center V2.0

- **SETUP steps:**
  – **DLIModel Utility**
    - Import IMS PSB and DBD source
    - Generate
      – IMS metadata .class file
  – **IBM Mashup Center**
    - Install metadata .class file
    - Create IMS DB feed via JDBC
    - Add feed to MashupHub Catalog



18

For IBM Mashup Center to access an IMS database through standard SQL queries, it needs information about the database. Because IMS databases are hierarchical, you must use the IMS Enterprise Suite DLIModel utility plug-in to translate IMS source files into metadata files. The metadata files describe the relational database view that IBM Mashup Center can use to generate SQL queries

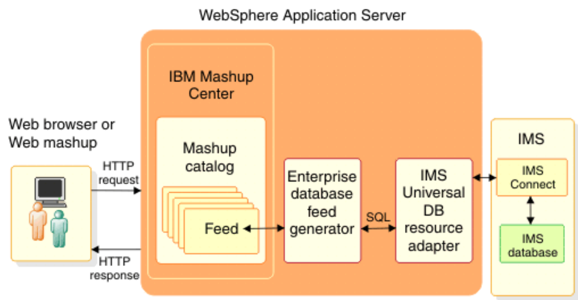
A feed service application developer first generates the metadata file from the IMS program specification block (PSB) and database description (DBD) source files. The generated metadata file (a .class file) needs to be compiled and copied into specific locations in IBM Mashup Center.

The developer then creates an IMS database feed in IBM Mashup Center by selecting Enterprise Database (JDBC) > IMS, and providing connection information. The feed is registered in the catalog.

IMS Version 11

Running an IMS 11 database Feed

- Invoking an IMS 11 DB feed
  - IBM Mashup Center Enterprise database feed generator
  - Mashup editing tools in IBM Mashup Center
  - IMS 11 Universal DB resource adapter



19

When the WebSphere Application Server receives an HTTP request for an IMS DB feed from a Web client, it passes the request to MashupHub which searches the catalog and invokes the IMS feed generator.

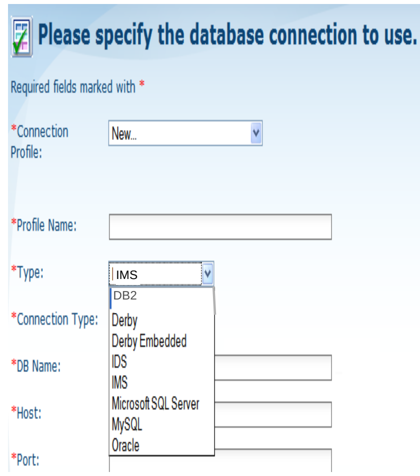
The IMS feed generator processes the parameters, establishes a connection with IMS Connect, and sends the request as an XML message.

The XML adapter function in IMS Connect converts the XML message into byte arrays by using the XML converter driver, and sends the data as an input message for the IMS transaction. The output byte array that is returned by the IMS transaction is converted by IMS Connect into XML response data by using the converter driver.

The response is then returned to the IMS feed generator, which converts the response to the Atom feed (XML Standard) format and sends it to the client as an HTTP response.

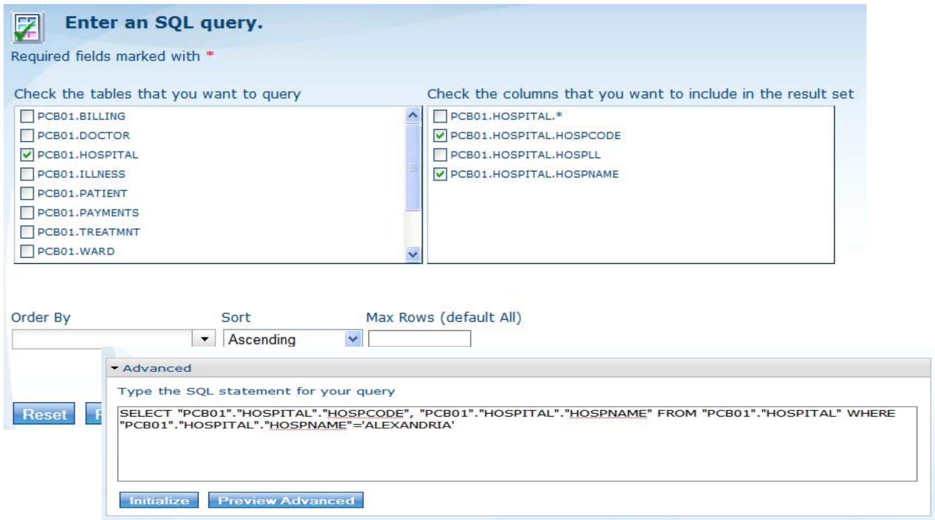
Support for IMS Data – Mashup Center (new slide)

- Enterprise Database Plugin
  - Allows users to create feeds from any Database that supports a JDBC driver
- Supports the following databases:
  - DB2, Derby, IDS, **IMS**



Support for IMS Data – Mashup Center…

- Supports creating SQL queries via GUI interface



IMS TM Resource Adapter 10.2

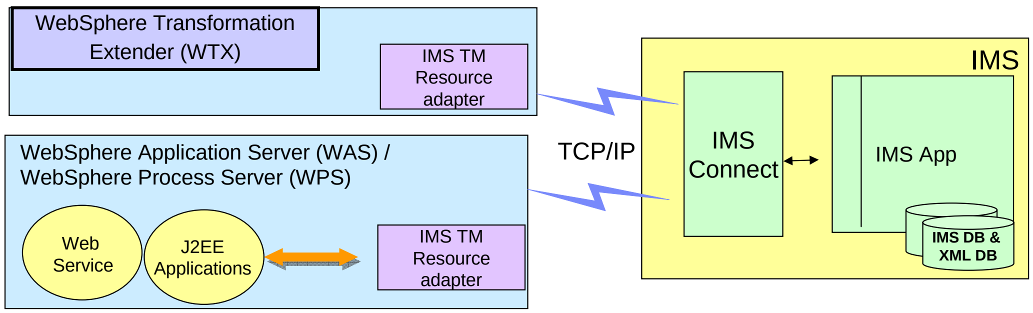
TM Resource Adapter – IMS 10 SPE

- WebSphere Transformation Extender (WTX)
- Socket Reconnect
- Send Only Reroute

WebSphere Transformation Extender (WTX) support

IMS Version 11

WebSphere Transformation Extender (WTX) support …



- Runs on
  - Microsoft™ Windows™, AIX®, z/OS Batch, z/OS IMS™, z/OS UNIX® System Services, Red Hat and SUSE Linux™, Solaris, and HP-UX.

25

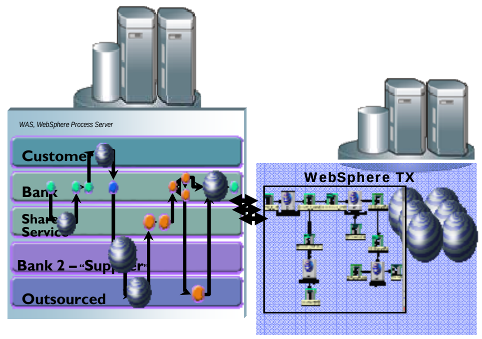
WebSphere Transformation Extender performs transformation and routing of data from source systems to target systems in batch and real-time environments. The sources may include files, relational databases, MOMs (message-oriented middleware), packaged applications, or other external sources. After retrieving the data from its sources, the WebSphere Transformation Extender product transforms it and routes it to any number of targets where it is needed, providing the appropriate content and format for each target system.

WebSphere Transformation Extender (WTX) is the new name for WebSphere DataStage TX (formerly Ascential DataStage TX) product. WTX is a powerful enhancement to the native transformation capabilities included in other WebSphere products. It can be deployed directly within other WebSphere products or as a stand-alone or embedded transformation engine. It provides:

- Flexible deployment options such as SOA, ESB, EDA, a stand-alone server, a business process orchestration, or programmatically through Enterprise JavaBeans (EJB), Javaä, C, or COBOL
- Consistent data transformation across the enterprise, independent of data structure, data location, infrastructure, and operating environment
- Reduced application development and maintenance costs with increased application deployment speed by reusing transformation assets
- Increased application quality by working in a code-free environment for transformation and validation of highly complex data
- Faster standards compliance and improved data quality with automated data validation using industry and regulatory standards
- Multiple execution options to support right-time, right-style transformation — batch, real time, or embedded
- Standards-based transaction support for unique industry transformation requirements such as X-12, EDIFACT, HIPAA, HL7, SWIFT, and NCPDP

WebSphere Transformation Extender (embedded edition)

- Runs on WAS, WPS, etc.
  - Allows WTX Maps to be programmatically executed through EJBs, Java, C/C++, COBOL programs
  - Relies on WTX APIs and SDK (Software Development Kit)
  - WTX Maps can be called as transformation services from WAS, WPS and WESB
  - WTX can be tightly integrated with customer applications and third-party software
    - WTX execution return code and status is returned to the calling client application
    - Transformation results can be returned to the calling client application or written to output by WTX adapters
  - WTX Maps can be called inline (synchronous) or asynchronously



WebSphere Transformation Extender (WTX) support

- Provides WTX access to IMS transactions via IMS TM RA
  - IMS TM RA supports
    - Send/Receive
    - Send only
    - Retrieving asynchronous output
    - Conversational transactions
    - Multi-segment output
- Software requirements
  - IMS and IMS Connect Version 10 or later
  - IMS TM Resource Adapter Version 10.2
  - WTX 8.2.0.2

27

WebSphere Transformation Extender (WTX) support for IMS invokes IMS transactions while leveraging standards-based transaction support on distributed platforms of complex data formats and unique industry requirements. By leveraging the IMS TM resource adapter, the following capabilities are supported:

Send/Receive

Send only

Retrieving asynchronous output

Conversational transactions

Multi-segment output

This support is for TCP/IP connections only

WTX Support - Benefits

- Benefits
  - Provides standards compliance access to IMS transactions
    - SEPA (XML format requirement for European banks)
    - X-12
    - EDIFACT (cross industry)
    - HIPAA (health care)
    - HL7
    - SWIFT (financial services)
    - NCPDP
  - Leverages the WTX support of complex data formats on distributed platforms

WTX support enhances the IMS participation in complex data formats for distributed platforms as well as the need to provide compliance to standards including SEPA, X-12, EDIFACT, HIPAA, HL7, SWIFT and NCPDP.

Socket Reconnect support

IMS Version 11

Socket Reconnect support

- Ability to re-establish a broken connection with IMS Connect
  - Without user intervention
  - invoked when IMS TM RA detects:
    - IOException or EOFException from TCP/IP

- Allows IMS Connect to be recycled transparently to the end-user IMS TM RA client

30

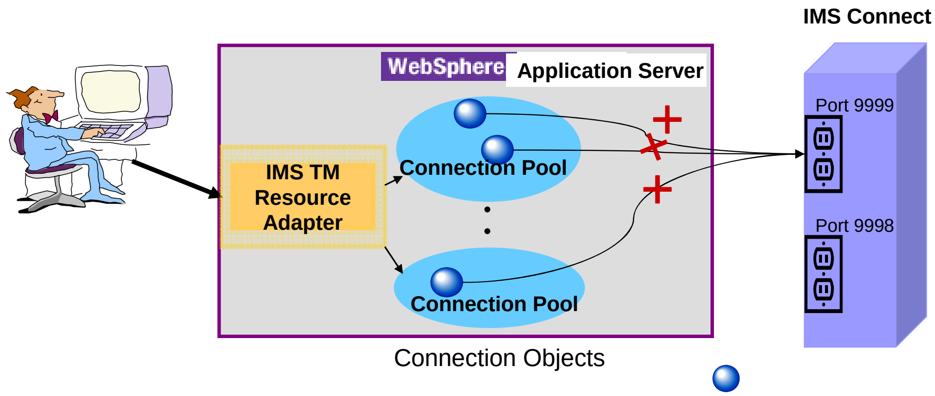
The socket reconnect function provides the capability to re-establish any stale connections in a connection pool whenever one of the connections in the pool encounters a communication failure in the process of sending a request to or receiving a response from IMS Connect. To avoid duplicate interactions being processed in IMS, the enhancement will only re-establish the connection but not retry any interactions.

If a TCP/IP IO error occurs IOException is generated. Additionally, an EOFException on a receive can occur because the socket has disappeared.

IMS Version 11

Socket Reconnect Support …

- Problem: If IMS Connect is recycled or experiences an error
  - All connections in pool become unusable (stale)
  - All connections in pool are tried before new connections are created
- Solution: Stale connections are automatically reconnected

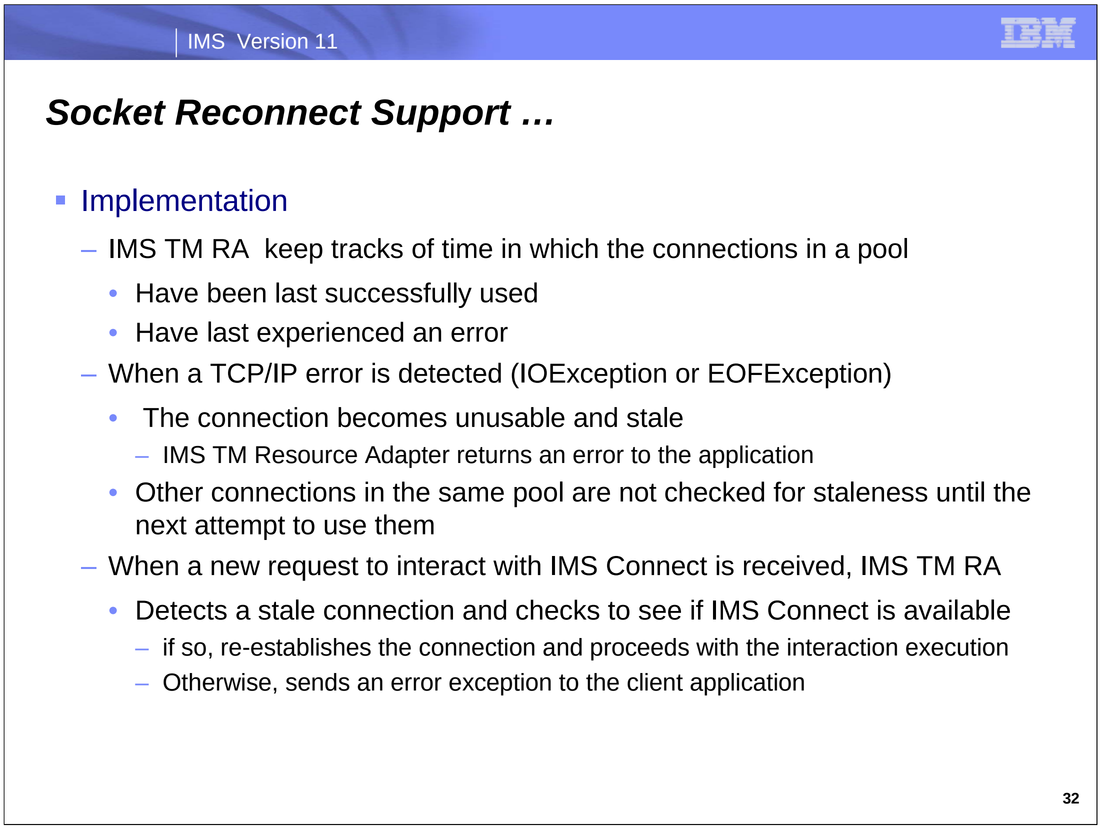


31

Each connection factory has a pool of connections with one or more connection objects. In a non-Sysplex Distributor environment, each object represents a connection to a port on the same IMS Connect instance. When that IMS Connect goes down, each of these connections becomes unusable or stale. As requests continue to be processed, every application that requests a connection is given a stale connection from the pool until there are no more stale connections in the pool. Each attempt to use a stale connection causes an exception to be thrown to the application indicating that the connection is unusable and must be removed from the pool. When all of the stale connections have been removed from the pool in this way, new connection objects are created and added to the pool for use in future interactions. This method of processing is problematic because of the performance impact as well as the added complexity to the IMS TM RA client programming to address the exceptions thrown.

What connection retry enhancement provides is the ability for IMS TM RA to catch the first exception and rather than pass it on to the application, IMS TM RA attempts to restore the unusable connection. Once the connection is restored, IMS TM RA executes the interaction so the application can continue as designed. Only if IMS TM RA fails to restore the connection does it throw an exception.

When Sysplex Distributor is used to route requests to multiple IMS Connects using the same host name and port number, the picture is a little more complicated, but the problem is fundamentally the same and has no impact on this version of connection retry.



The socket reconnect support provides a new mechanism to keep track of when the connections in a connection pool were last used successfully and when they last experienced an error. The error time is updated whenever a TCP/IP connection error condition, IOException or EORException, is detected.

When a request to send a message to IMS Connect is received, IMS TM RA checks to see the status of the connection. If it is stale, then an automatic function to check the status of IMS Connect is invoked. If IMS Connect is available, then the connection is restored and the interaction occurs. If IMS Connect is not availabe, the stale connection is destroyed and the IMS TM RA application receives an error condition.

Socket Reconnect Considerations

- Setup Considerations
  - To take advantage of the new functionality, the existing resource adapter in WebSphere Application Server must be replaced with the new adapter
  - Neither the IMS application nor IMS TM RA application require modification to make use of this functionality
- Restrictions
  - Socket Reconnect is supported for TCP/IP connections only
- Software requirements
  - IMS and IMS Connect 9, 10 or later
  - IMS TM Resource Adapter Version 10.2
  - WebSphere Application Server 6.0 or above (6.1 recommended)

33

To take advantage of this new capability, the WebSphere Application Server administrator will need to replace any existing IMS TM Resource Adapter with the new one. Note that neither the IMS application nor the IMS TM RA client require any modification. The enhanced support is transparent to the applications and is accomplished at the resource adapter layer.

Socket reconnect support only applies to the TCP/IP connections and not to the LOCAL support.

Socket Reconnect support

- Benefits
  - Reduces communication exceptions after IMS Connect is restarted
  - Enhances socket connection availability and fault tolerance
    - Allows applications seamless recovery from connection errors

This function allows IMS Connect to be recylced transparently to the IMS TM RA client and seamlessly let them continue with their interactions without encountering connection errors and having to resubmit them. Additionally, other connection problems that results in connection error exceptions being thrown can be retried.

Socket Reconnect support is of special value in the area of fault-tolerance where connections between IMS TM RA and IMS Connect could potentially go bad either due to known issues (like a deliberate IMS Connect recycle) or in some unknown scenarios where, for example, the connection could become invalid due to some temporary network glitch.

Send Only Reroute support

Send Only Reroute support

- Enhances the existing Commit Mode 0 (CM0) send-only support to define a reroute destination
  - Supports shareable persistent sockets
    - Providing a way to retrieve the IOPCB reply
- J2EE application developer
  - sets the interactionVerb to SYNC_SEND
  - sets reroute property in the IMSInteractionSpec to TRUE
  - sets 1-8 character reroute name in the IMSInteractionSpec
    - default value HWS$DEF
- Benefit
  - Simplifies Client Application programming
    - I/O PCB output created by Send-Only request is rerouted to specified name

36

IMS Transaction Manager Resource Adapter Send Only Reroute support reroutes output replies associated with Send-Only interactions for CM0.

To use the reroute function for Send only interaction, the Java application developer needs to

Set the interactionVerb to SYNC_SEND.

Set the reroute property in the IMSInteractionSpec to TRUE.

Specify a maximum 8-character Reroute name in the IMSInteractionSpec. The reroute name represents the name of the destination to which asynchronous output is to be rerouted and queued. If a Reroute name is not provided, the Reroute name in the IMS Connect configuration file will be used. If no value is specified in the IMS Connect configuration file, the default value HWS$DEF is used.

With the reroute flag set to True and the reroute TPIPE name specified, any IOPCB output message resulting from the Send-Only transaction is queued to the reroute TPIPE. This is of value to CM0 interactions on a shareable persistent socket because these types of interactions do not allow a user-specified clientid. The previous challenge therefore, when retrieving the subsequent IOPCB output, was that the remote client had no way to know what tpipe name to specify thereby restricting the ability to retrieve the reply. With this new capability, the client can define the tpipe name for IMS to use for the IOPCB output so that the subsequent resume tpipe request can identify the correct name or, at minimum, use the default name of HWS$DEF.

Generated Clientid support

Note: This capability was discussed in the IMS Connect Section and repeated here because it is IMS TM RA that takes advantage of the support

Generated Client ID

- Mechanism to request that IMS Connect generate a Client ID
  - Impacts ITMRA (IMS TM Resource Adapter) environments
    - For shareable persistent sockets
      - Where the client ID representing a unique socket/TPIPE must be generated rather than end-user specified
- Addresses duplicate client ID error
  - When multiple WAS instances are configured
    - Each ITMRA generates a unique client ID which may or may not be unique across multiple WAS instances

38

The IMS TM Resource Adapter (ITMRA) supports both dedicated and shareable persistent sockets. Dedicated persistent sockets only support CM0 (commit mode 0) interactions and require user-specified Client IDs that represent the socket connection and TPIPE. Shareable persistent sockets, on the other hand, can be shared (serially reused) by multiple applications running either CM1 (commit mode 1) or CM0 interactions.

For shareable persistent sockets, ITMRA does not allow the remote client to specify the client ID, rather it uses an algorithm to generate one for this type of interaction. Although the algorithm is coded to ensure that the value is unique within an ITMRA instance, the possibility exists for the same value to be generated by ITMRA across multiple WAS instances. If this happens then a request from ITMRA could be rejected with a duplicate client error status. To prevent this error condition, environments that have many instances of WAS running on distributed platforms use different IMS Connect ports. Such a setup requires the administration of the WAS configurations and many additional TCP/IP ports for IMS Connect.

IMS Version 11

Generated Client ID …

- Implementation details
  - New: Send Client ID field with a generated value and set flag OMUSR_F2_CIDREQ in byte OMUSR_FLAG2
    - If the value in the Client ID field is a duplicate
      - IMS Connect generates another client ID
        - Returns flag OMUSR_F2_CIDGEN in byte OMUSR_FLAG2 to indicate a generated clientid is being returned
- Benefit
  - Eliminates the requirement for different IMS Connect PORTs for instances of distributed WAS using ITMRA Shareable Persistent sockets
    - Simplifies WAS configuration and operations management

39

To address the situation, IMS Connect provides mechanisms that generate the Client ID for ITMRA shareable persistent sockets and ensures uniqueness so that all the instances of WebSphere can specify the same IMS Connect TCP/IP port.

The IMS Connect protocol provides two options:

- The first option is for a remote client to send a blank Client ID. The blank value tells IMS Connect to generate a unique Client ID to represent the shareable persistent socket.
- The second option, a new IMS Connect option and the one that ITMRA uses, allows ITMRA to generate a Client ID, as it has in the past, but with a flag setting of OMUSR_F2_CIDREQ in byte OMUSR_FLAG2. The flag tells IMS Connect to generate a unique clientid only when the provided clientid results in a duplicate client condition. In this second case, IMS Connect returns the newly generated Client ID to ITMRA on a reply message.

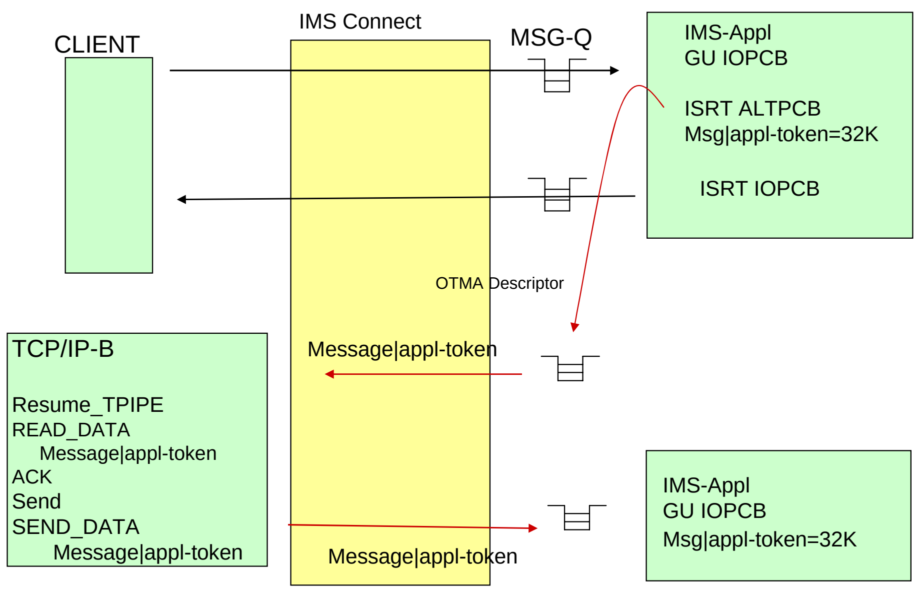
The generated Client ID support, therefore, ensures Client ID uniqueness (socket/TPIPE name on the IMS Connect/OTMA side) for each shareable persistent socket so that all the instances of WebSphere can specify the same IMS Connect TCP/IP port.

IMS 10 Synchronous Callout SPE

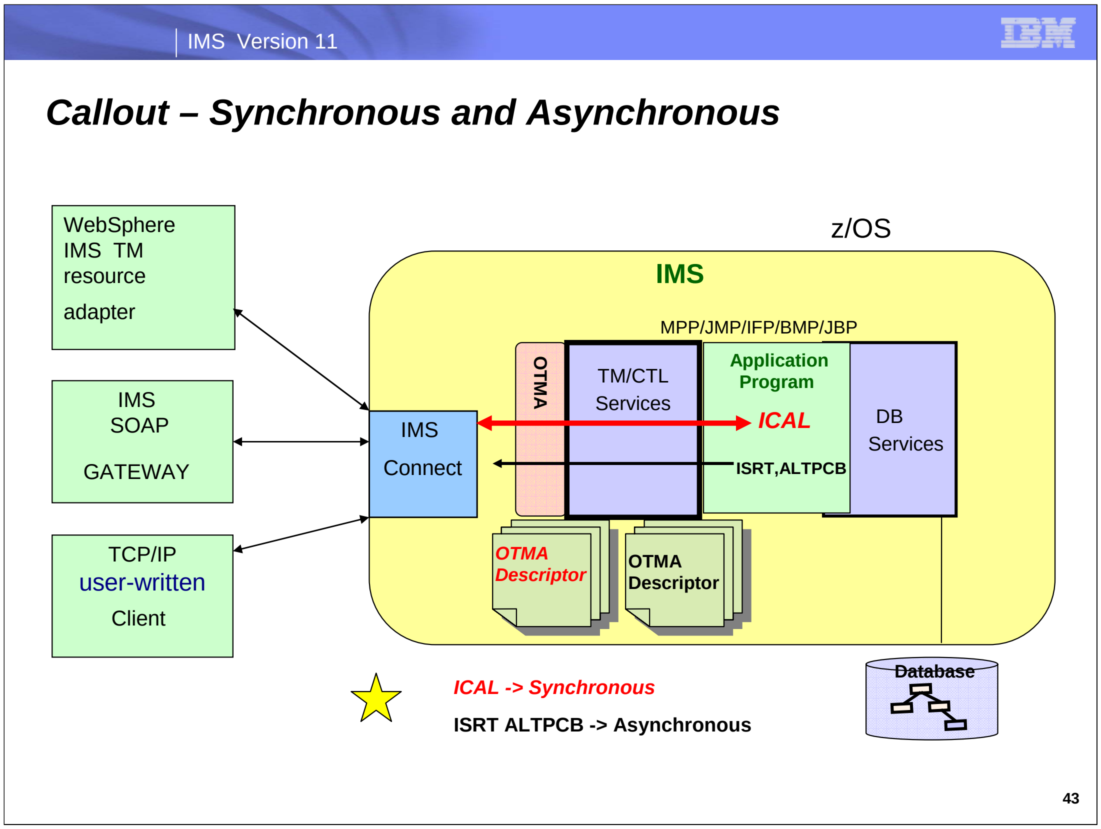
Callout

- Asynchronous Callout
  - Allows IMS transactions to access a service outside IMS
  - The IMS application does not wait for a reply
    - Any replies invoke a new instance of a transaction
- Synchronous Callout
  - IMS 10 SPE (APAR/PTF information at the end of this section)
  - Allows IMS transactions to access a service outside IMS ***and*** wait for a reply within the same unit of work
    - Position IMS as both a client and a server
    - Integrates IMS with other servers and applications
    - Removes application managed message correlation

Asynchronous Callout Review



42



This diagram shows that with the IMS callout support IMS applications can now function as both a client and server. The IMS Application Program can callout either synchronously or asynchronously to:

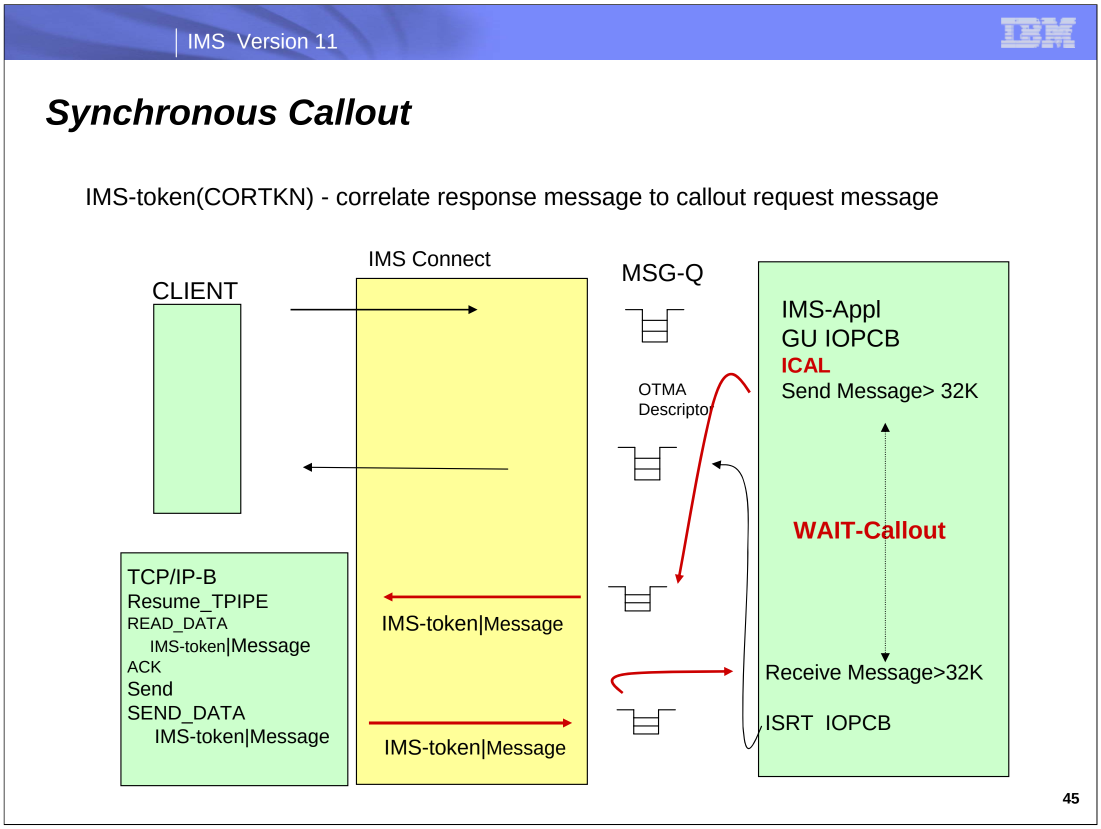
- User-written IMS Connect clients
- WebSphere EJBs/MDBs using theIMS TM Resource Adapter
- To any Web Service Provider using the IMS SOAP Gateway

Synchronous Callout Request

- New AIBTDLI call - ICAL
  - Provides the synchronous callout capability
  - Timeout support
    - Optional user specified timeout value in the DLI call to control the time an IMS application waits for a response
    - Can terminate the request and free the dependent region
    - Late responses are logged and discarded
  - Relieves the 32K segmentation limitation
    - IMS Connect and OTMA handle buffer and segmentation internally

44

Synchronous Callout line item enhances the current asynchronous IMS Callout support to allow the IMS application to callout synchronously and wait for the response to come back. Using this programming style, the IMS application program can initiate direct communication with other external application programs and receive the response in the same IMS transaction instance



The IMS-token (CORTKN) is a structure that used to correlate a response to a callout request with the IMS application program that issued the callout request.

Solution Details

- New DLI ICAL
- OTMA support
- IMS Connect support
- DFSDDLT0
- IMS TM Resource Adapter
- IMS SOAP Gateway
- Commands support
- Log Records

ICAL AIBTDLI Interface

Call AIBTDLI USING ICAL,aib,REQ_area,RESP_area

- ICAL is new DL/I call verb
  - SENDRECV is the new subfunction
- REQ_area is the Request data area for sending data
- RESP_area is the Response data area for returned data
  - REQ and RESP messages are not recoverable
  - req-area and resp-area do not specify LLZZ, data can be > 32K

47

The ICAL synchronous call function is only supported using the AIBTDLI interface in IMS TM runtime environments for IFP, MPP, BMP, JMP and JBP regions.

ICAL is the new function for call processing and SENDRECV is the new sub function for synchronous callout.

REQ_area is the request data area for sync callout. Note - do not specify LLZZ.

RESP_area is the response data area for returned data. Note - do not specify LLZZ.

ICAL AIBTDLI Interface . . .

- AIB
  - AIBID = DFSAIBbb
  - AIBLEN = AIB length
  - AIBSFUNC = SENDRECV
  - AIBRSNM1 = 8 byte OTMA Descriptor name
  - AIBRSFLD = Timeout value
    - 4 byte field for time value 100th seconds. System default is 10 sec.
  - AIBOALEN = **REQ_area** length
    - As an input parameter: 4 byte field contains the length of the request area
    - As an output parameter: Actual length of the response message
  - AIBOAUSE = **RESP_area** length
    - As an input parameter: 4 byte field contains the length of the response area
    - As an output parameter: Length of the response message.
  - AIBRETRN = AIB Return code
  - AIBREASN = AIB Reason code.
  - AIBERRXT = 2 byte sense code from external application

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.

Correlator Structure

▪ HWSIMSCB Macro

```
*************************************************************
*  CORTKN   Synch Callout Correlator Token
*************************************************************
CORMask     DSECT             Correlator Token dsect
COR_Len     DS   H            Length of COR Structure
COR_Rsvd    DS   H            Reserved
COR_Id      DS   CL8          Str Id *CORTKN* ascii/ebcdic
COR_LL      DS   XL2          Length of Token
COR_RESV1   DS   XL2          Reserved
COR_IMSID   DS   CL4          IMSID
COR_MEMTK   DS   XL8          Member XCF Token
COR_AWETK   DS   XL8          Message Token
COR_TPIPE   DS   CL8          TPIPE Name
COR_USERID  DS   XL8          Userid
CORMask_Len EQU  *-CORMask    Size of COR Structure
```

49

The Synchronous Callout Correlation Token is passed to the Resume TPIPE as COR Structure (defined in HWSIMSCB Macro). It is used by IMS to manage the IMS application program ICAL request and response correlation.

ICAL COBOL Interface

```
01 AIB.
   02 AIBRID PIC x(8)  VALUE 'DFSAIB  '.
   02 AIBRLEN PIC 9(9) USAGE BINARY.
   02 AIBRSFUNC PIC x(8) VALUE 'SENDRCV'.
   02 AIBRSNM1 PIC x(8) VALUE 'DFSISOAP'.
   02 AIBOALEN PIC 9(9) USAGE BINARY VALUE +12.
   02 AIBOAUSE PIC 9(9) USAGE BINARY.

01  CALLOUT-MSG.
   02  CA-DATA        PICTURE X(12) VALUE ‘HELLO WORLD ’
01  SCA-RESPONSE.
   02  SCA-DATA       PICTURE X(12).

CALL 'AIBTDLI' USING ICAL, AIB, CALLOUT-MSG , SCA-RESPONSE.
```

50

COBOL sample

ICAL Java Message Service (JMS) Interface

- IMS JMP/JBP

```
com.ibm.ims.jms.IMSQueueConnectionFactory;
setTimeout(1000);
setMaxOutputLength(128000);
createQueue("OTMA Descriptor name");
request("Hello World");
replyMsg Hello IMS
```

51

The Java Message Service (JMS) API is a messaging standard for sending messages between two or more application programs. IMS supports the point to point model for ICAL.

ICAL REXX Interface

Address REXXIMS ← Setup REXX environment to use IMS REXX function

Input = ‘Hello World’
Timer = 6000
Descriptor = ‘OTMDEST1’

'SET SUBFUNC SENDRECV' ← Set the sub-function code
'SET RSNAME1 Descriptor' ← Set descriptor name
'SET TIMER Timer' ← Set timer

'ICAL DFSAIB Input Output' ← Make ICAL

Say Output ← Display output

Hello IMS

OTMA Destination Routing Descriptors

- DFSYDTx member of IMS.PROCLIB
  - TYPE: Destination type
  - TMEMBER: OTMA Target Client
  - TPIPE: Destination Name
  - SMEM: YES|NO
  - ADAPTER: Type of IMS Connect Adapter
  - CONVERTR: Routine called by Adapter
  - **SYNTIMER=timeout**
    - **Note ICAL overrides this value**

IMS 11 Type -2 Command

```
UPDATE OTMADESC NAME(OTMASYN) SET(SYNTIMER(5000))
```


IMS OTMA Destination Routing Descriptors externalize the routing definitions and specifications for callout messages without IMS user exits. They are read and initialized at IMS startup. The maximum number of routing descriptors is 256.

IMS 11 provides a new Type-2 UPDATE OTMADESC command that be used to dynamically change the SYNTIMER value in the Descriptor.

For example, entering a command of: “UPDATE OTMADESC NAME(OTMASYN) SET(SYNTIMER(5000))” updates an existing SYNTIMER value to 5000 from the value of 2000 for the OTMASYN descriptor which could have been previously specified in the DFSYDTx descriptor as:

D OTMASYN TYPE=IMSCON TMEMBER=HSW2 TPIPE=HWS2SOAP

D OTMASYN ADAPTER=XMLADPTR CONVERTR=XMLCNVTR SYNTIMER=2000

Client Programming – Flow

RESUME TPIPE Commit Mode=0 SyncLevel=Confirm

LLLL|LLRRIRM|0400|

Callout request message includes IMS CORRELATOR TOKEN(CORTKN)

LLLL|*CORTKN*|LLRRCORTOKEN|{*REQMOD*}|LLLLDATA*CSMOKY*

Reply ACK or NAK

LLLL|LLRRIRM|0400|

Send Response Message

Callout response message for IMS application program

Include IMS CORRELATOR TOKEN(CORTKN)

LLLL|IRM|LLRRCORTOKEN|LLLLDATA|0400|

54

For Synchronous Callout support, IMS generates a correlation token and includes it in the outgoing callout request automatically. The IMS Connect client must ensure that the correlation token is also included in the response that is returned to IMS. The presence of the correlation token in the response message identifies it as a synchronous callout response message to OTMA and IMS Connect.

Client Programming – IMS Connect

Commit Mode 0 Resume TPIPE IRM provides 3 options:

1. Asynch-only - default (existing clients without changes)
2. Synch-only – Client can only process synchronous callout
3. Synch + Asynch – Client can process both types of callout

Client Programming – Callout Message Format

- Callout message format indicates Asynchronous or Synchronous

Asynch:

LLLL |{*REQMOD*}| LLZZData | {LLZZData} |*CSMOKY*

Synch:

LLLL | *CORTKN* |{*REQMOD*}| LLLLData |*CSMOKY*

An IMS Connect client application program can be written to support both asynchronous as well as synchronous callout requests.

After determining the LLLL value, the client application can be coded to check for the values *REQMOD* and *CORTKN*. If the field contains the value of *CORTKN* the application program will need to process the callout request synchronously. For synchronous interactions, the data is not segmented and can therefore be >32K. Additionally, the new CORTKN structure must be returned by the client application program as part of the response messsage to the Synch Callout Request.

Client Programming – Synch Callout Send-Only Response

- Client send-only response

IRM_ARCHITECTURE LEVEL = 3
IRM_FLAG4 SET TO SYNRESP ('M') Sync Call- Send only response
IRM_CORTKN SET TO IMS PROVIDED CORRELATOR TOKEN
RESPONSE MESSAGE

- Client send-only error response

IRM_ARCHITECTURE LEVEL = 3
IRM_FLAG0 SET TO NAK WITH REASON
IRM_NAK_RSNCDE FIELD SET IN AIBERRXTN
IRM_FLAG4 SET TO SYNRESP ('M')
IRM_CORTKN SET TO IMS PROVIDED CORRELATOR TOKEN
Client application error response data

57

Setting the IRM_ARCHITECTURE LEVEL to 3 in the IRM prefix specifies that the response message contains the CORTKN correlation token fields.

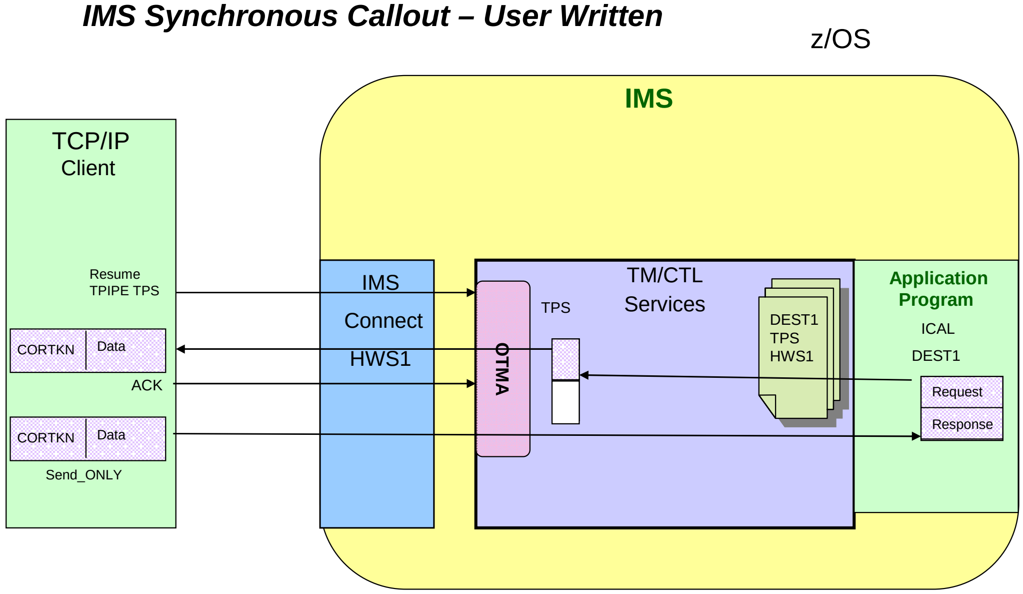
The client application can also create a send-only error response by setting the IRM_FLAG4 SET TO SYNRESP ('M'). This capability is of value in situations when the client application has accepted the Synch Callout (replied ACK) and then subsequently determines that the request is invalid or the required resources are not available. The client application can return an error message with an extended error code to the IMS Application ICAL. The error response will also include the message text (application data).

Client Programming – NAK Synch Callout

- NAK-STOP
  - Reject message and terminate Resume TPIPE
  - AIBRETRN=X'100' AIBREASN=X'108' return to ICAL
    - Client can provide extended error code for AIBERRXT
- NAK-Continue (Reroute)
  - Reject message, continue retrieving messages for Resume TPIPE
  - AIBRETRN =X'100' AIBREASN=X'108' returned to ICAL
    - Client can provide extended error code for AIBERRXT
- NAK-Pause (Hold Message)
  - Terminate this Resume TPIPE and keep message
    - OTMA will hold message for another Resume TPIPE request
    - Used if message is OK but don't want to receive more messages
    - Note ICAL is waiting for response message

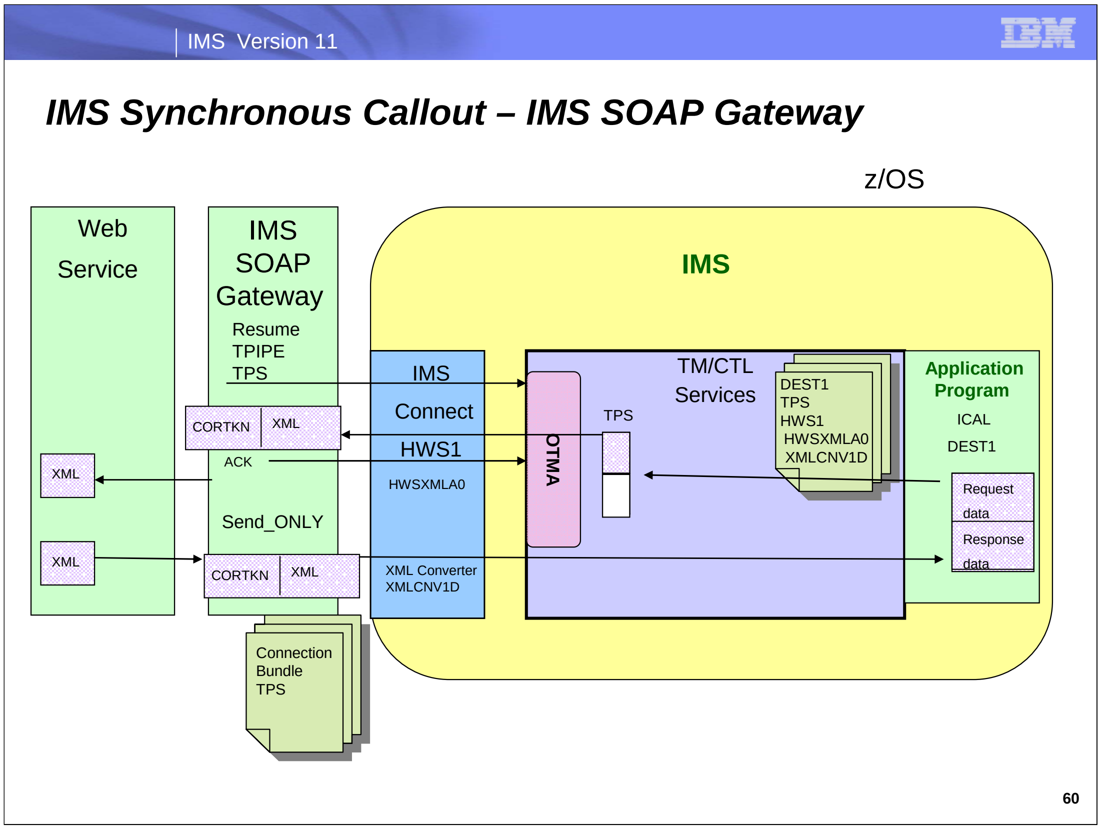
IMS Connect provides remote client with other options to control interaction during error scenarios.

IMS Synchronous Callout – User Written



59

User-Written applications can use the enhanced Resume TPIPE function to retrieve the callout request and use the Send-Only function to send back the response. A correlation token will be generated by IMS for the callout request message. This token will be sent to the application together with the callout request message inside the IRM header. The application needs to send back the same correlation token in the IRM of the response message to IMS Connect.



The IMS SOAP Gateway can be easily configured to wait for synchronous callout messages. The connection bundle in the IMS SOAP Gateway is where the connection and security properties between IMS SOAP Gateway, IMS Connect, and IMS can be defined.

At some point, during runtime, an IMS application issues the ICAL call to send the callout request data using an OTMA descriptor name and optional timeout value. As a result of the ICAL, a correlation token is sent together with the callout request. OTMA starts the timer to wait the response message and the IMS application synchronously waits for the response.

When the message gets to IMS Connect, the IMS Connect XML Adapter loads the XML converter XMLCNV1D to convert the request data from bytes to XML. It also adds Web service correlation information in the message. IMS Connect delivers the request message to IMS SOAP Gateway.

The IMS SOAP Gateway receives the request message and immediately sends an ACK to IMS Connect to acknowledge the message. IMS SOAP Gateway looks up the callout correlator and the WSDL file based on the Web service correlation information in the callout request message. The outbound SOAP request to the actual remote service will be built based on the correlation and WSDL file information to invoke the external web service provider. At some point, the Web Service provider sends a response back to IMS SOAP Gateway which send it to IMS Connect.

The IMS Connect XML Adapter loads the XML converter XMLCNV1D to convert the response data from XML to bytes. It sends the response message back to OTMA. IMS correlates the message back to the corresponding IMS transaction instance and the IMS application program ICAL request completes.

IMS SOAP Gateway Migration

- Existing asynchronous callout correlator file supports synchronous callout
  - The “transaction code” in the correlator file is ignored.

```
<?xml version="1.0" encoding="UTF-8"?>
<correlator:correlator
  xmlns:correlator="http://www.ibm.com/IMS/Correlator">
  <correlatorEntry operationName="IMSPHBKOperation"
  serviceName="IMSPHBKService">
    <adapterType>IBM XML Adapter</adapterType>
    <converterName>IMSPHBKD</converterName>
    <connectionBundleName>dfwims10</connectionBundleName>
<trancode>IVTNO</trancode>
    <calloutConnBundleName></calloutConnBundleName>
    <calloutWSDL></calloutWSDL>
    <calloutWSTimeout>7500</calloutWSTimeout>
  </correlatorEntry>
</correlator:correlator>
```

61

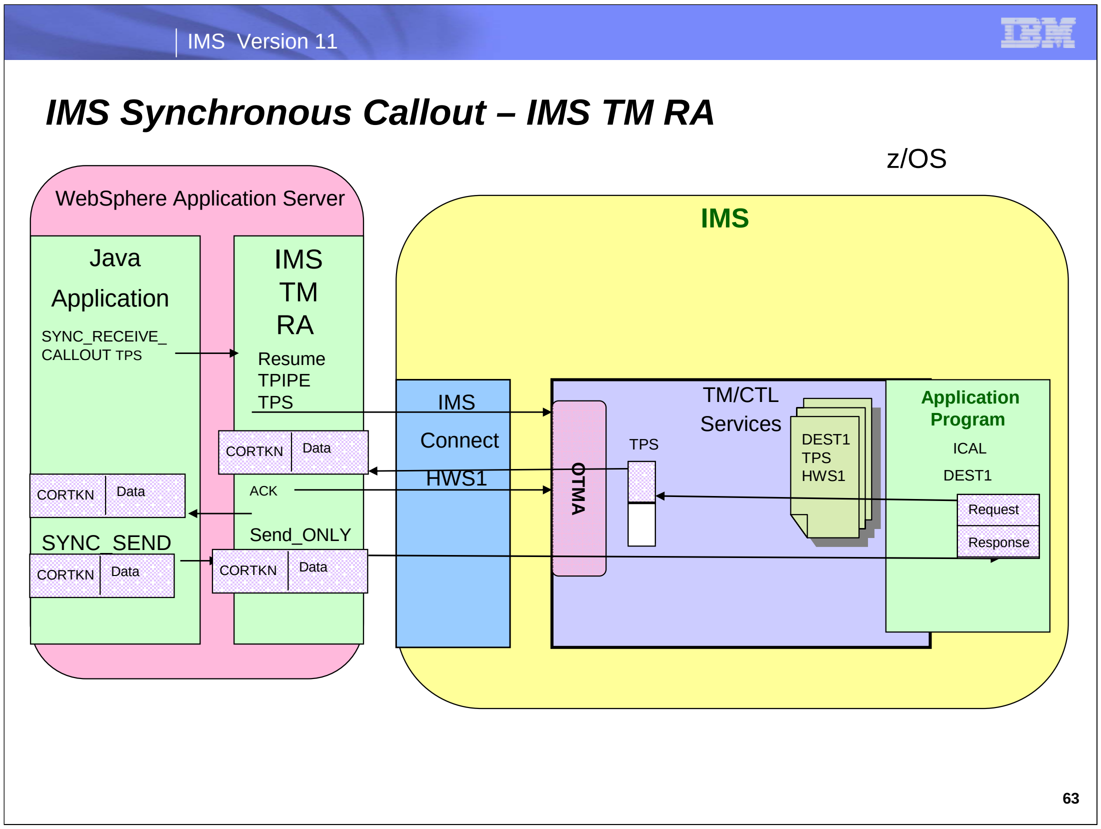
If a customer would like to use an existing asynchronous callout correlator file to support synchronous callout function, then the “transaction code” in the correlator file is ignored.

IMS SOAP Gateway Restrictions

- 32MB maximum XML message size for the request and response data
- PL/I applications are not supported until RDz 7.6

For IMS SOAP Gateway interactions , the maximum XML message size for both the request and response for synchronous callout messages is 32 MB for COBOL applications. This restriction is dependent on the version of Enterprise COBOL compiler that you configure to use in Rational Developer for System z. Enterprise COBOL compiler Version 3.4 allows a maximum XML message size of 32 MB, whereas a lower version of the compiler might allow only 16 MB.

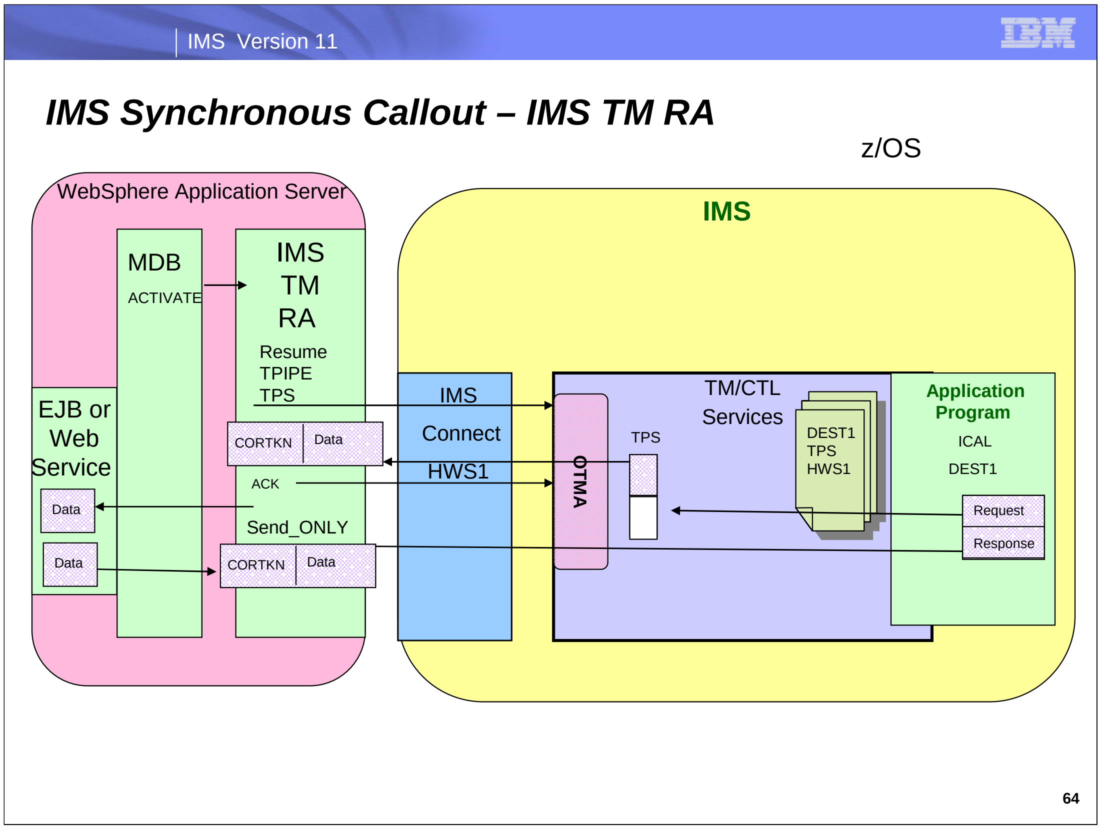
For IMS SOAP Gateway, PL/I applications are not supported until RDz. 7.6.



This visual shows what occurs when using the IMS TM Resource Adapter in a JEE environment.

For synchronous callout, an IMS application issues the DL/I ICAL call to send the request data with a specification of the OTMA descriptor name to be used and, optionally, a timeout value. OTMA delivers the request message to IMS TM Resource Adapter via IMS Connect. A correlation token is sent together with the callout request. IMS Connect delivers the request message to IMS TM Resource Adapter. After the send is successful, OTMA starts the timer to wait for the response message. The IMS application synchronously waits for the response.

The IMS TM Resource Adapter receives the request message and immediately sends an ACK to IMS Connect to acknowledge the message. It then delivers the callout request and the correlation token to the Client Managed Java Application. In the Client Managed programming model, the Java Application has the responsibility of executing the enhanced Resume TPIPE and Send-Only protocol and managing the correlation token. To do so, the Java Application uses IMS TM Resource Adapter Resume TPIPE interaction(SYNC_RECEIVE_CALLOUT) to retrieve callout from a specified TPIPE TPS. The correlation token is passed to the Java Application as one of the properties in the IMSInteractionSpec class called the SyncCalloutToken. The Java Application then has the responsibility to maintain the correlation token and pass it back to the IMS TM Resource Adapter when sending the response. By using the Send-Only interaction (SYNC_SEND) of the IMS TM Resource Adapter, the Java Application can return the response message and correlation token. IMS Connect receives the response, sends it to OTMA, and IMS correlates the message back to the corresponding IMS transaction instance.



The IMS Transaction Manager Resource Adapter JCA 1.5 inbound support provides standards-based support for message-driven beans if MDBs are to be used. JCA 1.5 specifies a message inflow contract for pulling asynchronous output and callout messages from IMS transactions. The IMS TM Resource Adapter manages the correlation token internally and does not pass it to the MDB application.

The flow is as follows:

The MDB is configured with an ActiviationSpec that contains the IMS Connect hostname, port number and TPIPE name information that is used by IMS TM Resource Adapter to retrieve the callout requests. The IMS TM RA waits for callout requests on TPIPE TPS.

When the IMS application issues the DL/I ICAL call to send the callout request data using the OTMA descriptor name, OTMA delivers the request message to IMS TM Resource Adapter via IMS Connect. A correlation token is also sent with the callout request. The IMS TM Resource Adapter receives the request message and immediately sends an ACK to IMS Connect to acknowledge the message. The IMS TM Resource Adapter then delivers the callout request to the MDB by calling the onMessage() method. The MDB processes the callout request, and can invoke EJBs or Web Service providers for further processing.

When the response is available, the MDB returns the response message to IMS TM Resource Adapter which in turn sends it back to the IMS using the Send-Only function via IMS Connect. IMS correlates the message back to the corresponding IMS transaction instance and the IMS application ICAL function completes.

Enhanced IMS commands

- /DIS TMEMBER TPIPE
  - WAIT-S (WT-S): The tpipe is waiting for ACK/NAK for a synchronous callout message
- /DIS TMEMBER TPIPE SYNC
  - Displays the detailed sync callout message count and status
- /DIS ACTIVE
  - WAIT-CALLOUT: New status for region waiting on synchronous callout response

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    1 MPP1A     TP    APOL11    APOL1    WAIT-CALLOUT         1
      JMPRGN    JMP   NONE
      JBPRGN    JBP   NONE
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
      DBTRGN    DBT   NONE
      DBRZCSAJ  DBRC
      DLIZCSAJ  DLS
*08213/165100*
```

65

The OTMA command, /DIS TMEMBER TPIPE is enhanced to include a new wait flag (WAIT-S) to indicate a particular TPIPE is waiting for a client for an ACK/NAK on a synchronous callout message.

The /DISPLAY TMEMBER TPIPE SYNC command is used to display:
- the number of active synchronous callout messages
- the number of synchronous callout messages waiting for response
- the resume tpipe option,
- the resume tpipe mode,
- the tpipe queue counts
- the tpipe status.

The /DIS ACTIVE REGION command response now includes TMEMBER as well as TPIPE. You can use the /DIS TMEM xx TPIPE yy SYNC to find status

Enhanced IMS commands . . .

- /PSTOP
  - Clear the wait in the region
  - Dequeues synchronous callout messages from the TPIPE
- /STOP TMEMBER TPIPE
  - Clear state of all messages for the TPIPE
- /STOP OTMA
  - Clear all the ICAL messages for all the TPIPEs.
  - New ICAL synchronous callout requests rejected
- These commands return to ICAL
  - AIBRETRN = 100
  - AIBREASN = 10C

When the /STOP OTMA command is issued, it will clear or reject all the ICAL messages for all the transaction pipes. /PSTOP, /STOP OTMA, or /STOP TMEMBER TPIPE, may be used to end the ICAL,
In this case, the IMS application will get return code 100 and reason code 10C.

Enhanced Messages

- DFS231
- DFS569I
- DFS1190I
- DFS1193I

IMS Connect ICAL information

- IMS Connect Recorder Trace
  - first 670 bytes of messages
- IMS Connect Events

Several IMS messages have been enhanced to support ICAL information.

In IMS Connect, you can trace synchronous callout requests and messages by using the IMS Connect Recorder Trace facility. When the IMS Connect Recorder Trace facility is activated, IMS Connect takes a snapshot of the first 670 bytes of messages at key points during IMS Connect processing.

IMS OTMA Log Records ICAL information

- X'6701'
  - Synchronous callout request data and response data
    - OTMA message prefix information and the first 256 bytes of data
  - Synchronous callout data is not recoverable across IMS restarts
- X'670D'
  - Error Conditions
    - NAK from OTMA client is received for the synchronous callout request
    - Timeout action was taken
    - NAK is sent to OTMA client for an incorrect response

IMS Monitor ICAL information

- X'4E'
  - IMS Monitor event log record.
- SLOG Codes
  - X'70' – DL/I systems call start
  - X'71' – DL/I systems call end
  - X'78' – Synchronous callout start
  - X'79' – Synchronous callout end

A simplified OTMA 6701 log record with the OTMA message prefix information and the data will always be cut for every synchronous callout data and its response data.

OTMA uses the 67D0 log records to log error conditions. For synchronous callout requests and their responses, OTMA cuts a 67D0 log record in the following conditions:

- When an ICAL call cannot be processed or sent to IMS Connect
- A timeout action was taken
- A NAK is sent to the OTMA client for an incorrect response

When the IMS monitor is activated, a X'4E' monitor record is created for the ICAL call. This record contains statistical information about the system. (DFSMNTR0)

SLOG codes are added to support the synchronous callout function.

IMS Log Records ICAL Information

- X'07'
  - Log record accounting data
    - Number of ICAL synchronous calls
- X'56FA'
  - Log record accounting areas
    - Number of ICAL synchronous calls

KBLA Option 4.1 Report.

```
-07 RECORD    APPLICATION PROGRAM TERMINATED
    PSB NAME: BMP255       TRAN CODE:              JOB NAME: INSERTDB
    REGION #: 0001         RCVY TOKN: C9D4E2F14040404000000000100000001
    REG TYPE: BMP REGION                           STEP NAME: BMP
    DATE/TIME: 2007-03-12 21:29:30.669179 UTC          LOG SEQ NO: 000008
    ** STATISTICS **
     # MSGS :      0 DB GU  :      0 DB GN  :      0 DB GNP :      0
     …..
     RLSE   :      0 XQ SAVE:      0 XQ RSTR:      0 XQ COPY:      0
     ICAL   :      0
```


The IMS x'07' Log record accounting area includes AIBTDLI ICAL call counts.

Prerequisites

- Software requirements
  - IMS and IMS Connect Version 10
    - PK70078/UK40363/UK40364, PK70330/UK40215, PK73224/UK40813 - Precondition
    - PK71135/UK42415/UK42416, PK74168/UK42459 – Activation
    - PK75209 for IMS Java dependent regions ICAL support
  - IMS 11 – PK85023/UK48124 for IMS Java ICAL support
  - For synchronous callout to J2EE application/Web Service in WebSphere Application Server (WAS):
    - IMS TM Resource Adapter Version 10.3
    - WAS 6.0 for callout to EJB
    - WAS 6.1 for callout to MDB
  - For synchronous callout to Web Service using IMS SOAP Gateway:
    - IMS SOAP Gateway Version 10.1.1

70

The synchronous callout support requires IMS and IMS Connect 10 with the SPE. APAR PK70078 and PK73224 contains all the IMS changes (e.g. IMS Systems, OTMA) and APAR PK70330 contains all IMS Connect changes. Activation APARS are PK71135 and PK74168.

The callout programming models for Java applications to retrieve synchronous and asynchronous callout requests from IMS applications have the following requirements:

If the synchronous callout request is issued from an IMS application in the IMS Java™ dependent region, also apply APAR PK75209 for IMS 10 and APAR PK85023 for IMS 11.

For callout to WebSphere application using IMS TM Resource Adapter, an updated IMS TM Resource Adapter is required. Additionally, WAS 6.0 or later is required for the Client Managed programming model and WAS 6.1 or later is required for the MDB programming model.

For use with the IMS SOAP Gateway, IMS SOAP Gateway Version 10.1.1 is required.

Prerequisites

- Tooling
  - For synchronous callout to J2EE application/Web Service in WAS
    - Rational Application Developer (RAD) V7.007 or later
      - V7.5 is recommended, (V7.6 for PLI)
      - IMS TM RA ifix 10.3
    - WebSphere Integration Developer (WID) V6.1 or later
    - MDB generation requirement to RAD/WID
  - For synchronous callout to Web Service using IMS SOAP Gateway
    - Rational Developer for System z (RDz) 7.1.1 or later

RAD V7.5 or WID 6.1 or later tooling is required to develop your Java Application code.

RDz Version 7.007 or later tooling is required. V7.5 is recommended and V7.6 for IMS PLI application structions.

Restrictions

- Synchronous callout does not support:
  - Shared queue environments where the IMS front-end and back-end systems are different when processing the request
    - The response message of a synchronous callout request must flow back to the originating IMS in order to be processed correctly (front-end must equal back-end)
  - OTMA Input/Output Edit exit routine (DFSYIOE0)
  - BMP or JBP applications running in a DBCTL environment
  - Two phase commit (distributed sync point)

72

The synchronous callout support will be provided in phases.

The first phase delivered as an IMS 10 SPE does not support Shared Queues environments configured where the processing would occur on a back-end IMS. Synchronous callout requires the front-end and back-end systems to be the same.

The OTMA Input/Output Edit exit routine (DFSYIOE0) is not supported for either synchronous callout request messages received by OTMA from IMS application programs that issue the DL/I ICAL call or synchronous callout response messages. Additionally, it cannot be used to update or cancel synchronous callout messages or responses.

Two phase commit is not be available in the first deliverable.

BMP or JBP applications running in a DBCTL environment are not supported.

The callout processing inside the IMS application must be single-threaded. That is, the IMS application can only issue one synchronous callout call at a time. It has to wait for the response (or timeout) before it can issue the next callout request.

Multiple resume tpipe requests with specification of an alternate client ID for the same tpipe will be ignored by OTMA and treated as one resume tpipe request.

IMS 10 SPE – Synchronous Callout

- Benefits
  - Position IMS as both a client and a server
  - Integrate IMS with other server and applications
  - Remove application managed message correlation
  - Removes 32K message segment restriction

73

IMS synchronous callout enhances the current asynchronous callout capability to allow IMS application to wait for the response to be returned in the same IMS transaction instance.

IMS SOAP Gateway 10 SPE Multi-Segment Messages

Multi-Segment support with IMS 11 is discussed in the IMS Enterprise Suite section

IMS Version 11

Multi-Segment Messages

- Capabilities
  - Support IMS application program multi-segment messages
    - Messages only formatted by the IMS Connect XML adapter
      - The IMS Application program cannot build XML messages
    - Asynchronous callout is not supported
    - Tooling only available for COBOL applications
- Prerequisites
  - Software requirements
    - IMS Connect 10 SPE APAR PK69366/UK52038
    - IMS SOAP Gateway 10.1
  - Tooling
    - Rational Developer for System z 7.5.1

75

The IMS SOAP Gateway can send and receive multi-segment messages for the IMS as a “provider” scenario. This provides IMS multi-segment applications the ability to participate in SOA from Windows, AIX, Linux on System z, and z/OS environments, using the IMS Connect XML Adapter function.

Multi-Segment Messages

- Benefits
  – SOAP Clients can now use IMS SOAP Gateway for access to IMS multi-segment applications

Open Database

.

IMS Version 11

The Open Database support allows access to IMS DB from programs running not only locally to IMS (same LPAR)

But now also

from programs in a different LPAR or even from remote TCP/IP environments

- Open Database Capabilities
  - Supports open-standards for connectivity to online IMS databases
  - Provides an environment that manages access to online IMS databases
  - Provides Open Database APIs
    - Ease application development access to IMS databases

78

IMS Open Database

- supports open standards that can be used for distributed and local connectivity to IMS databases
- manages access to databases owned by IMS DB systems.
- provides a new application programming called Open Database API to ease the development of application programs that access IMS databases from many different distributed and local environments.

Open Database

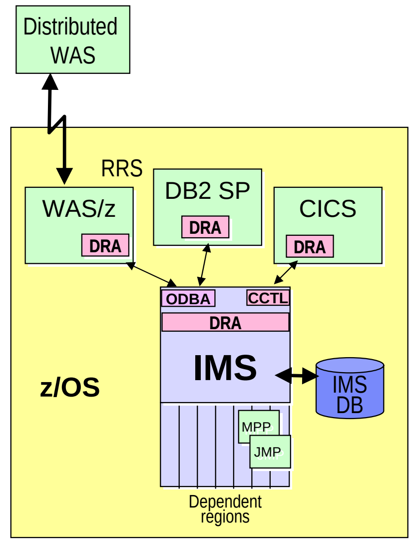
- Review of DB Access in IMS 9/ IMS 10
- New Opportunities in IMS 11
- Open DB Solutions
- IMS Universal Drivers
- Implementation Details
  - Installing Universal Drivers in WAS
  - ODBA Enhancements
  - ODBM Implementation
    - Setup and operation
  - IMS Connect
  - Non-JDBC Client APIs

IMS Version 11

Ways of Accessing Online IMS DB in IMS 9/10

- (Non-java) Dependent Region (MPP or BMP)
- Java Dependent Region (JMP or JBP)
- (Non-java) ODBA (requires RRS)
  - e.g. COBOL DB2 Stored Procedure
- Java ODBA (requires RRS)
  - e.g. WAS/z or Java DB2 Stored Procedure
- (Non-java) CCTL
  - e.g. CICS DBCTL
- Java CCTL
  - e.g. Java CICS DBCTL
- Distributed WAS (requires WAS/z and RRS)

CICS is syncpoint co-ordinator



80

Existing IMS 9/ IMS 10 interfaced to access IMS DB include:

- Open Data Base Access (ODBA) which is used by environments such as WAS/z and DB2 SP. ODBA requires the enablement of MVS/RRS as the syncpoint coordinator.
- CICS uses the CCTL interface and, because CICS functions as the syncpoint coordinator, does not impose a requirement for RRS.
- Both ODBA and CCTL use the Database Resource Adapter (DRA) interface. Some IMS modules reside in the ODBA application or CICS address spaces, most notably the DRA Startup Table (Assembled DFSPRP macro).
- IMS Java provides two IMS DB “Classic” Drivers. One for Java dependent regions, CICS, and DB2 Stored Procedures (supports the use of DLI calls or JDBC SQL), and one for JEE Environments (the JDBC Resource Adapter).
- IMS 9 introduced the IMS Java Remote Data Services (RDS) capability which allows a program running in a distributed WAS environment to communicate with IMS-provided helper EJB components running on WAS/z to relay DL/1 request to IMS via ODBA.

Problems with the IMS 9/ IMS 10 Solutions

- If an ODBA application prematurely terminates, there is the possibility of crashing IMS with a 113 abend
  - If application terminates in the middle of a DL/1 call
- When used, WAS/z must run in same LPAR as IMS
  - Can effect machine capacity and licence charges
- Remote WebSphere solutions always additionally require WAS/z
  - “helper” EJB running in WAS/z relays remote request to ODBA
- Distributed applications accessing IMS DB, must execute on WAS
  - No support for other distributed platforms

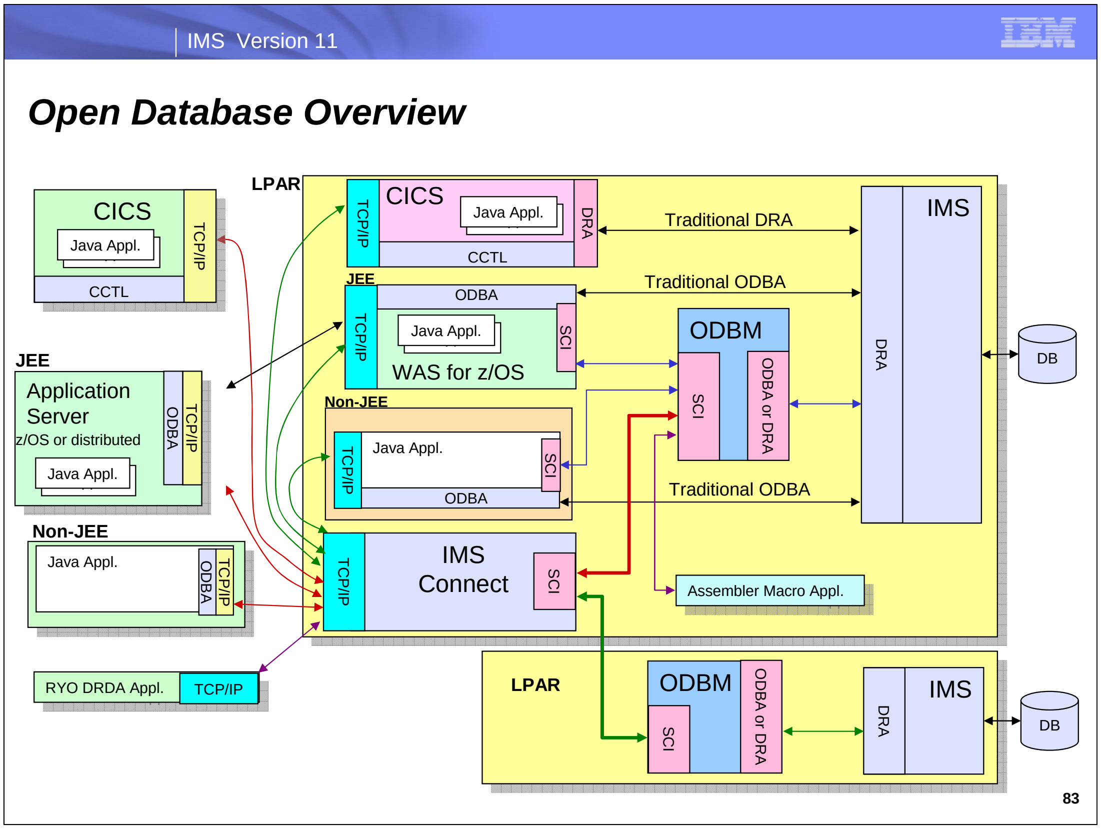
81

Even with all the capabilities, however, remote access to IMS DB continued to have several issues and considerations.

Open Database

- Open Database consists of three components:
  - Open Database Manager (ODBM)
  - IMS Connect
  - Open Database API and the Universal Drivers

Three main components combine to provide Open Database access to IMS Databases. We will describe the Open Database Manager and it’s environment. We will describe the changes to IMS Connect to support TCP/IP access to the Open Database Manager environment and the Open Database APIs that can be used to access IMS DB via the Open Database Manager environment.



As this visual shows, IMS Version 11 enhances IMS Connect, along with a new IMS common service layer (CSL) component called Open Database Manager (ODBM), to support open standards for connectivity to IMS databases. Database access, therefore, can be requested not only from local applications on the same LPAR (existing Type2 connectivity) but can now cross LPAR boundaries or even be sent in from remote environments (new Type4 connectivity). Together, IMS Connect and ODBM provide a Distributed Relational Database Architecture (DRDA) server capability for IMS. IMS Connect uses the structured call interface (SCI) to communicate with ODBM and has access to all the ODBM address spaces in the same IMSplex environment.

Open Database

- Terminology
  - IMSplex
    - One or more IMS systems working together as a single unit
  - Common Service Layer (CSL)
    - The infrastructure needed for IMSplex systems and resource management
  - Coordinator Controller (CCTL)
    - Transaction programs that use the DRA to access online IMS databases
  - Database Resource Adapter (DRA)
    - An interface to IMS HALDB, full-function and DEDB databases
  - DRA Thread
    - Structure connects a CCTL task or an ODBA task with an IMS DB task
  - Open Database Access (ODBA)
    - Application programs that use the DRA to access online IMS databases

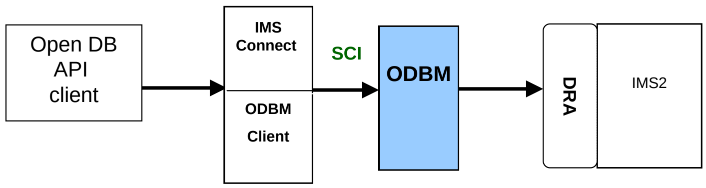
Open Database

- Terminology
  - Open Database Manager (ODBM)
    - CSL address space manages connections to online IMS databases
  - ODBM Client
    - Programs that access ODBM such as IMS Connect
  - Open Database API
    - Provides Universal Drivers for direct access to IMS databases
  - Type 2 Local Connection
    - z/OS environment same LPAR IMS database access
  - Type 4 TCP/IP Connection
    - Network and across LPAR distributed environments IMS database access
  - Universal Drivers
    - Support Type 2 and 4 connections for Java application programs

Open Database

- Software requirements
  - IMS Version 11
  - IMS Connect
    - Required for type 4 Universal driver connectivity
  - ODBM
    - Required for type 4 Universal driver connectivity
    - Required for type 2 ODBA to ODBM compatibility
  - Java 2 Technology Edition, v5.0 or later
  - Resource recovery services
    - Required for type 4 two phase commit
      - Not required for type 4 one phase commit
    - Required for the following type 2 runtimes
      - WebSphere Application Server on z/OS
      - DB2 z/OS stored procedures
    - Required for ODBA to ODBM compatibility

The ODBM Component



Open Database Manager

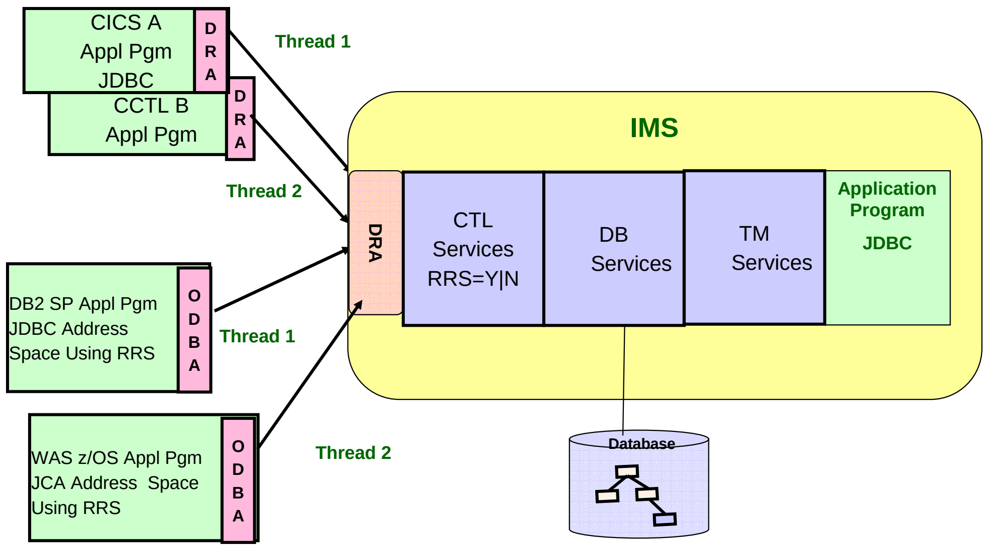
- Open Database Manager Capabilities
  - Common Service Layer address space
    - Manages connections to online IMS databases
    - Can be used to reduce ODBA Client action U113 abends
    - Implements the DRA interface
    - Supports Two-phase commit semantics
      - RRS=Y RRS provides sync point coordinator role
    - Supports Single-phase commit semantics
      - RRS=N ODBM provides CCTL sync point coordinator role

88

IMS Open Database Manager (ODBM) is a CSL address space that manages connections to databases owned by IMS DB systems. ODBM can be used to reduce the impact of an ODBA client failure that can result in an IMS U113 abend. For access to online IMS databases, ODBM supports the DRA interface used by a Coordinator Controller (CCTL) and Open Database Access.

Since IMS DB is a participant in the two phase commit process, ODBM can provide the CCTL syncpoint coordinator function and can use RRS as the syncpoint coordinator for the ODBA interface.

CCTL, ODBA and DRA Application Review



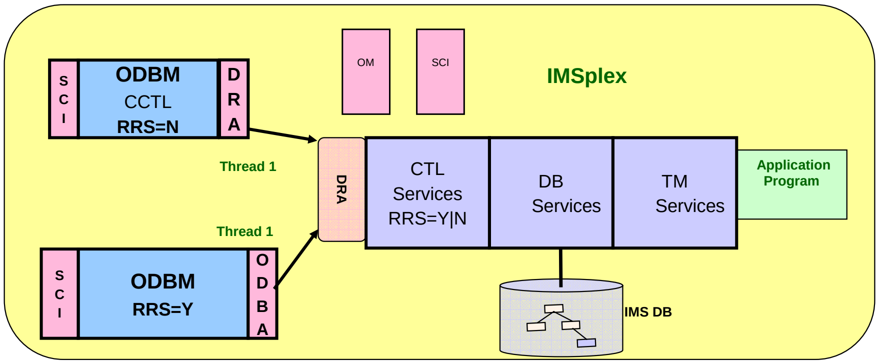
89

The Database Resource Adapter (DRA) is an interface to IMS DB full-function databases and data entry databases (DEDBs) that can be used by a coordinator controller (CCTL) or a z/OS® application program that uses the Open Database Access (ODBA) interface.

A DRA thread is a DRA structure that connects a CCTL task or a z/OS application program task with an IMS DB task that can process those calls. A single DRA thread is associated with every CCTL or ODBA thread.

Note: When ODBA is used, ODBC requires RRS which means that IMS must also use RRS.

ODBM Interfaces: ODBA and DRA

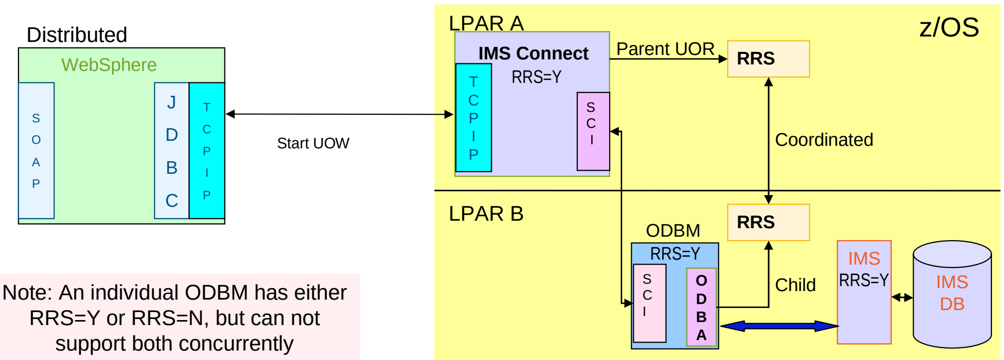


- ODBM can be started with either RRS=Y or RRS=N
- RRS=Y
  – Use ODBA
- RRS=N
  – Use CCTL

The CCTL interface does not require RRS and the ODBA interface does require RRS. ODBM supports both the DRA and ODBA interfaces but a single ODBM instance cannot support both. If the ODBM CCTL interface uses DRA then RRS=N is specified, otherwise for the ODBA interface, RRS=Y must be specified.

More on Syncpoints and RRS

- Use of RRS with ODBM is optional (ODBM start-up parameter)
  - RRS=Y is needed to supported Distributed Two-Phase Commit
    - RRS=Y in IMS Connect and ODBM and IMS Control Region
- If RRS=Y (default), ODBM uses the ODBA interface (i.e. AERTDLI)
- Otherwise, for RRS=N, ODBM uses the CCTL DRA interface



Note: An individual ODBM has either RRS=Y or RRS=N, but can not support both concurrently

91

RRS=Y must be specified in the configuration for a global transaction supporting two-phase commit.

This visual shows how Open Database achieves cross LPAR transaction management. When a client establishes a connection through IMS Connect to ODBM, several things are done to establish a coordinated Unit of Work. First IMS Connect creates the parent Unit of Recovery. IMS Connect then sends the UR token to ODBM which ultimately expresses interest in the UR as a child. At this point coordinated Unit of Recovery has been established.

Supports Connectivity to Multiple IMS DB systems

- New CIMS CONNECT call which allows ODBA applications to connect to multiple IMS subsystems with a single call
  - Previously CIMS INIT was the only function available, which only allowed the caller to connect to a single IMS
    - Multiple CIMS INIT calls were required to connect to multiple IMS DB systems
  - Implemented by ODBM

92

The ODBA interface has a new command for IMS Version 11: CIMS CONNECT. This command initializes the ODBA interface and supports connectivity to multiple IMS DB systems. Prior to IMS Version 11, ODBA applications used the CIMS INIT command to initialize the ODBA interface and connect to a single IMS DB system. If the application wanted to connect to multiple IMS DB systems, it had to issue multiple CIMS INIT commands, one for each IMS DB system. The ODBA interface has been enhanced in IMS Version 11 to support a new command, CIMS CONNECT, which allows the ODBA application to initialize the ODBA interface and connect to multiple IMS DB systems with a single command.

Coexistence –ODBM can access IMS 9 / 10/ 11

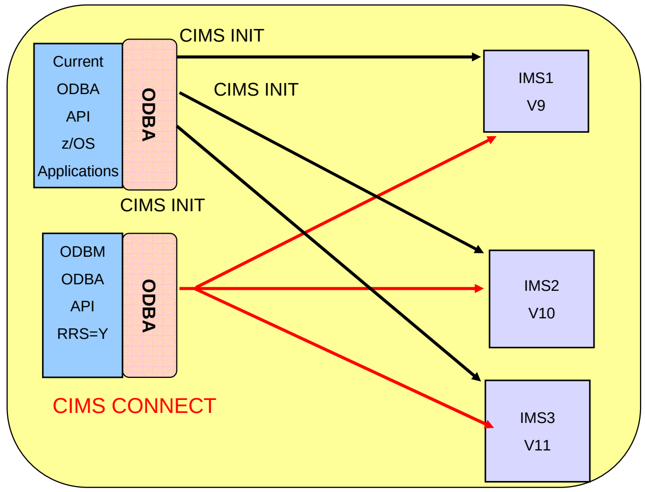
New CONNECT subfunction

Used by ODBM Address Space

Call AERTDLI parmcount, CIMS, AIB

parmcount = set to n (optional)
AIB = Address of AIB

- AIBSFUNC=CONNECT (required)
- AIBRSA1 = A(Parm list) (required)



Coexistence APARS

For IMS Version 9: PK66020/UK42176
For IMS Version 10: PK66022/UK42410

93

The ODBA interface from previous versions of IMS can coexist with IMS Version 11 without modification.

The IMS 9/ IMS10 ODBA coexistence APARs are required to allow an ODBM to connect to an IMS 9/ IMS10 subsystem.

During processing of the DATASTORE parameter in the CSLDCxxx member, but before any ODBM clients connect to ODBM, the ODBM address space uses the new ODBA CIMS CONNECT support. The coexistence APARS provide the supporting code necessary in the IMS core ODBA modules in IMS 9/ IMS10.

The CIMS CONNECT call can be used by existing or new ODBA application programs. The CIMS INIT call requires a separate call to each IMS.

And … for Compatibility

- ODBA application programs can use the ODBM address space
  - "**Compatibility Mode**"
  - Gives protection from potential U113 abends when ODBA applications are stopped during DLI processing
  - No changes required to ODBA applications
  - Need to update and recompile DFSPRP macro to build new DRA Startup Table

94

Existing ODBA applications can also use ODBM to protect IMS from abends that are caused by the unexpected termination of the ODBA applications during DL/I processing. ODBM support for ODBA application programs requires no changes to the application program, but does require minor changes to the DFSPRP macro, which defines the connections to IMS DB, and a recompile and rebinding of the DFSxxxx0 load module. You can modify your existing ODBA application servers to use ODBM by adding the IMSPLEX and ODBMNAME keywords to the DFSPRP macro. After adding these keywords, you must recompile and rebind the DFSxxxx0 load module (xxxx is the DRA start-up table name specified on the APSB call in the AIBRSNM2 field of the AIB).

ODBA Compatibility Support

DFSPRP Macro

FUNCLV=2

DBCTLID=

MAXTHRD=

MINTHRD=

.

.

.

IMSPLEX=

ODBMNAME = (Optional)

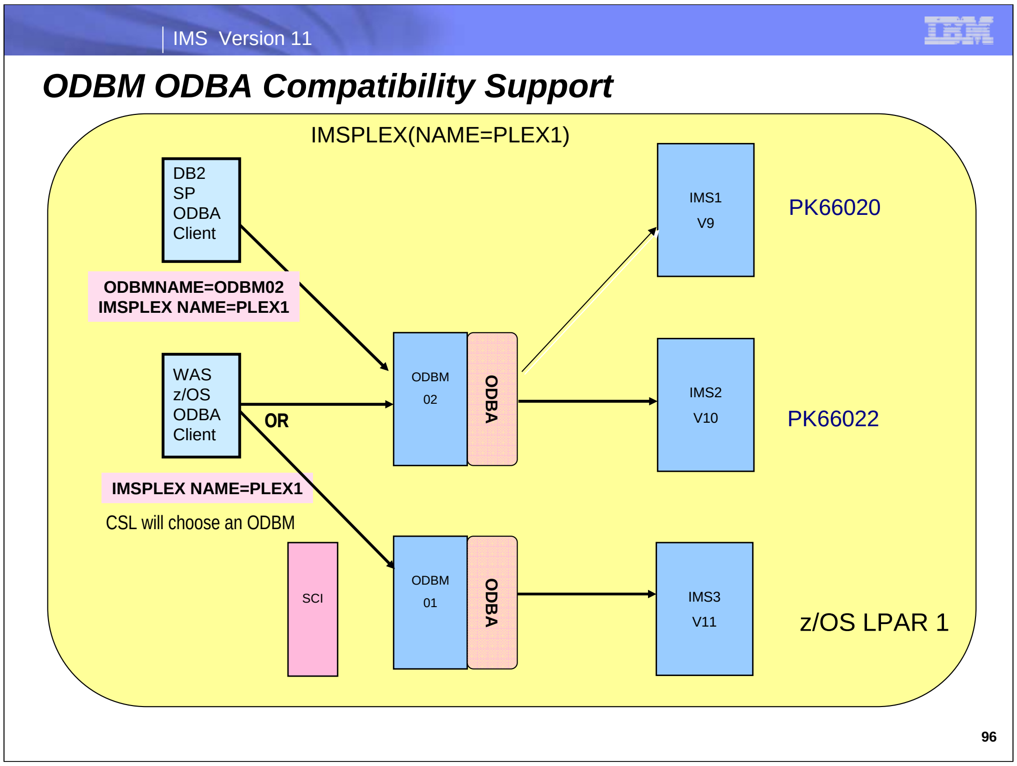
Causes ODBA requests to be routed via ODBM

```
//DFSIVP10 EXEC PROC=ASMDRA,MBR=DFSIMSA0
//ASM.SYSIN DD *
DFSIMSA0 CSECT
         DFSPRP DSECT=NO,                                        X
            FUNCLV=2,                ODBA FUNCTION LEVEL          X
            DDNAME=DFSDB2SP,         DDNAME FOR DRA RESLIB        X
            DSNAME=IMS110P.SDFSRESL, DSNAME FOR DRA RESLIB        X
            DBCTLID=IMSA,            DBCTL IDENTIFIER             X
            USERID=,                 USER IDENTIFIER              X
            MINTHRD=1,               MINIMUM NUMBER OF THREADS    X
            MAXTHRD=1,               MAXIMUM NUMBER OF THREADS    X
            TIMER=60,                IDENTIFY TIMER VALUE DEFAULT X
            FPBUF=,                  NUMBER OF FP BUFFERS PER THREAD X
            FPBOF=,                  NUMBER OF FP OVERFLOW BUFFERS X
            SOD=A,                   SNAP DATASET OUTPUT CLASS    X
            TIMEOUT=60,              DRATERM TIMEOUT VALUE        X
            IDRETRY=0,               IDENTIFY RETRY COUNT         X
            CNBA=,                   TOTAL FP NBA BUFFERS FOR CCTL X
            IMSPLEX=PLEXA,           IMSPLEX NAME                 X
            ODBMNAME=OD0A
         END
//*
```

95

Existing ODBA application servers can be modified to use ODBM by adding the IMSPLEX and ODBMNAME parameters to the DFSPRP FUNC level 2 macro. This compatibility support enables the use of ODBM by existing ODBA application programs without requiring application code modifications. In this environment the use of ODBM can help reduce U113 abends caused by ODBA application program failures.

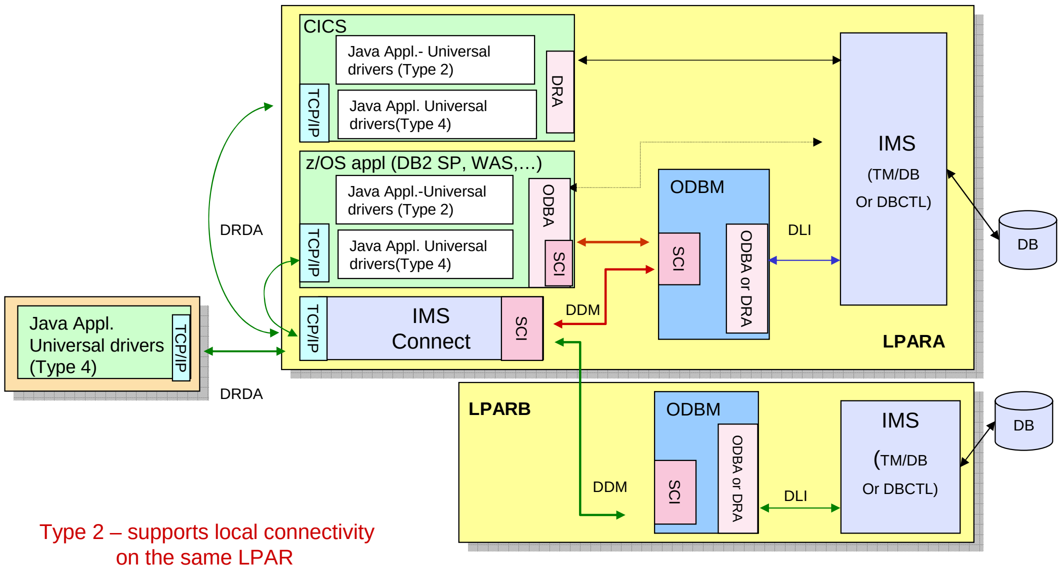
The IMSPLEX NAME= is used by the IMS V11 ODBA interface to register with SCI. ODBMNAME= specifies the name of the ODBM to which ODBA registers. If ODBMNAME= is not specified, ODBA will select the ODBM to route all ODBA calls using the IMSplex SCI. After adding these parameters, you must recompile and rebind the DFSxxxx0 load module.



This slide shows ODBA compatibility support and the IMS coexistence APARs to provide connectivity. The compatibility support does not allow crossing LPAR boundaries.

This example shows an existing WAS/zOS client and an existing DB2 Stored Procedure using the ODBA compatibility support without modifying the DB2 SP or WAS/z application code.

ODBM Connectivity



Type 2 – supports local connectivity on the same LPAR

Type 4 – supports remote access as well as crossing LPAR boundaries

97

This slide shows the capabilities provided by Open Database inclusing support for Type 2 connections for Local(same LPAR as IMS) access to IMS as well as Type 4 connections for distributed access to IMS. Note distributed access can be across a network or across an LPAR boundary.

ODBM – Setup

- ODBM Initialization Member
  – CSLDIxxx
- ODBM Configuration Member
  – CSLDCxxx
- ODBM BPE Configuration PROCLIB Member
  – BPECFG=
- ODBM BPE Managed User Exit List PROCLIB Member
  – EXITDEF=
- ODBM Execution Parameters
  – Specifies CSLDIxxx, CSLDCxxx and BPECFG for ODBM startup
  – Can be used to override CSLDIxxx Parameters

98

ODBM uses two IMS.PROCLIB members:

CSLDIxxx member is used to specify parameters that initialize the ODBM address space.

CSLDCxxx member is used to configure the ODBM data store connections to IMS systems.

BPECFG - The ODBM component is built on the Base Primitive Environment (BPE) common system service base and uses the BPE PROCLIB member to configure it's BPE execution environment parameters.

EXITDEF - The EXITDEF statement in the BPE PROCLIB member specified by EXITMBR specifies the ODBM exit routine types and modules to be managed by BPE.

ODBM Initialization Member CSLDIxxx

- Parameters for initialization of the ODBM address space.
  - ARMRST=Y|N
    - Specifies MVS Automatic Restart Manager (ARM) used to restart the ODBM
  - RRS= Y | N
    - Y - Specifies ODBM registers with RRMS and use the ODBA interface
    - N - Specifies ODBM is to use the CCTL/DRA interface
  - ODBMNAME
    - Specifies the name for the ODBM address space
    - Must supply a unique 1-6 character name
    - Each instance of an ODBM address space must have a unique ODBMNAME
    - ODBMID is the ODBMNAME followed by the characters "OD".
      - Used as ODBM Member name for IMSplex Group
  - ODBMCFG=000 | xxx
    - Suffix for the ODBM configuration PROCLIB member, CSLDCxxx

99

If you specify the ODBMNAME parameter in the CSLDIxxx PROCLIB member, you must also either define a separate CSLDIxxx member for each instance of ODBM in an IMSplex or specify ODBMNAME in the startup procedure for each instance of ODBM in the IMSplex.

ODBM Initialization Member CSLDIxxx . . .

- IMSPLEX(NAME=plexname)
  - Specifies the IMSplex for ODBM to join
  - Required parameter.
  - Only one IMSPLEX keyword may be specified
  - plexname
    - A 1-5 character value identifies the XCF CSL IMSplex group name.

QUERY IMSPLEX NAME(plexname) TYPE(ODBM) STATUS( ) SHOW( )

ODBM Initialization Member CSLDI001
Example 1 – Default ODBA interface

```
**********************************************************************
* CSLDI001 ODBM CSL PROCLIB MEMBER                                   *
**********************************************************************
ODBMNAME=ODBMB1       /* ODBM id = ODBMB1OD */
ODBMCFG=B11           /* Suffix for CSLDCxxx member */
IMSPLEX(NAME=PLEX1)  /* XCF group = IMSPLEX1)      */
```

ODBM Initialization Member CSLDI002
Example 2 – Set to CCTL DRA interface

```
**********************************************************************
* CSLDI002 ODBM CSL PROCLIB MEMBER                                   *
**********************************************************************
ODBMNAME=ODBMB2       /* ODBM id = ODBMB2OD */
ODBMCFG=B11           /* Suffix for CSLDCxxx member */
RRS=N                 /* Use CCTL/DRA              */
IMSPLEX(NAME=PLEX1)  /* XCF group = IMSPLEX1)      */
```

101

In Example1, ODBMNNAME=ODBM01, RRS does not need to be specified since RRS=Y is default which also implies the use of the ODBA interface to access IMS DB.

In Example 2, ODBMNAME=ODBM02, the desired use of the CCTL/DRA interface requires the specification of RRS=N.

Note both ODBM01 and ODBM02 can use the same CSLDCxxx (configureation) proclib member as shown in the next set of visuals.

ODBM Configuration Member CSLDCxxx

- Datastore connection initialization parameters

  The global parameters are specified under the label:

  <SECTION=GLOBAL_DATASTORE_CONFIGURATION>

  - IDRETRY=0 | nnn (maximum 255)
    - Number of times ODBM will retry connect to an IMS
    - If first connect attempt fails
  - TIMER=60 | nn (minimum 1, maximum 99)
    - Time (in sec) between attempts by ODBM to establish connectivity to IMS
  - MAXTHRDS=1 | nnn (maximum 999)
    - Number of concurrent active threads (A thread connects an ODBM with IMS)
    - Note review IMS MAXPST value
  - FPBUF=0 | nnn (maximum 999)
    - Number of Fast Path DEDB buffers allocated and fixed per thread
  - FPBOF=0 | nnn (maximum 999)
    - Number of Fast Path DEDB overflow buffers allocated per thread
  - CNBA=0 | nnnn (maximum 999)
    - Total Number of Fast Path NBA buffers for ODBM's use

102

The Global_Datastore_Configuration provides the common parameters that can be used for 1 or more Local_Datastore_Configuration definitions

- IDRETRY: the number of times ODBM will try to connect to an IMS data store if the first attempt is unsuccessful
- TIMER: the amount of time to wait between connection attempts
- MAXTHRDS: the number of concurrent active threads that ODBM can use to access an IMS system. Note: An ODBM thread will use an IMS PST for DL/I call processing. Review the MAXPST specification. If the ODBM maxthrds value is not reached but IMS maxpst value is reached, the database connection request is rejected and the end client is informed with a call failure.

ODBM regions accessing Fast Path resources must specify the number of buffers to be allocated as normal buffers (NBA)

- CNBA: the total number of Fast Path normal buffer allocation (NBA) buffers that an instance of ODBM can use for all of its IMS data store connections
- FPBUF: the number of Fast Path DEDB buffers to allocate and fix per thread
- FPBOF: the number of Fast Path DEDB overflow buffers to allocate per thread

ODBM Configuration Member CSLDCxxx . . .

The local IMS Datastore parameters are specified under the label:

<SECTION=LOCAL_DATASTORE_CONFIGURATION>

- ODBM
  - (Name= Name of the ODBM address space
  - corresponds to ODBMNAME= in CSLDIxxx
- DATASTORE
  - (Name= datastore name (i.e. IMS name)
  - 1-4 characters
- ALIAS
  - (Name=IMS alias name
  - 1-4 characters
  - Required parameter Optional for client to use.
- Optional local parameters (overrides global parms)
  - FPBUF=
  - FPBOF=
  - CNBA=
  - MAXTHRDS=

NAME= corresponds to ODBMNAME= in CSLDIxxx proclib member to identify the ODBM instance that is to use the DATASTORE statements.

- A datastore is equivalent to IMS subsystem ID value
- An alias name is an optional value that can be used by the client application program to access an IMS datastore

Multiple alias names can specify the same datastore name. An alias name and a datastore name can be the same value.

IMS Version 11

ODBM Configuration Member CSLDCB11 Example

```
********************************************************************
* CSLDCB11 ODBM CSL PROCLIB MEMBER                                 *
********************************************************************
<SECTION=GLOBAL_DATASTORE_CONFIGURATION>
IDRETRY=5              /* Retry connection 5 times before quit      */
MAXTHRDS=10            /* 10 threads max to any IMS Datastore       */
TIMER=30               /* 30 seconds between ID retry attempts      */
FPBUF=10               /* 10 DEDB buffers per thread                */
FPBOF=10               /* 10 Overflow buffers per thread            */
CNBA=200               /* (FPBUF*MAXTHRDS) + FPBOF <= CNBA          */
/******************************************************************/
/* Define DATASTORE properties for ODBM01                          */
<SECTION=LOCAL_DATASTORE_CONFIGURATION>
ODBM(NAME=ODBMB1,                                  /* Define parms for ODBM01  */
    DATASTORE(NAME=IMS1,                           /* IMSID on LPAR A          */
              ALIAS(NAME=IM1A,NAME=IO1A),          /* Names for APPL sets 1 & 2*/
              FPBUF=0,FPBOF=0,CNBA=0  ) )          /* No FastPath on this IMS  */
    DATASTORE(NAME=IMS2,                           /* IMSID on LPAR A          */
              ALIAS(NAME=IM1A,NAME=IO2A),          /* Names for DEDB apps      */
              FPBUF=50,FPBOF=50,CNBA=500,          /* FastPath on this IMS     */
              MAXTHRDS=5 ) )                       /* Throttle down threads    */
/******************************************************************/
/* Define DATASTORE properties for ODBM02                          */
ODBM(NAME=ODBMB2,                                  /* Define parms for ODBM02  */
   DATASTORE(NAME=IMS3,                            /* IMSID on LPAR B          */
              ALIAS(NAME=IO3A,NAME=IMS3) )         /* Names for APPL sets 3 & 4  --- take default values for the rest of the options*/
   DATASTORE(NAME=IMS4,                            /* IMSID on LPAR A          */
              MAXTHRDS=5 ) )                       /* Throttle down threads    */
```

A note on ALIAS names
- Another name to specify an IMS datastore
-discussed in greater detail in the IMS Connect section

104

Note LOCAL section can override values specified in GLOBAL section and one ODBM can be defined to access multiple IMS systems.

A note on ALIAS names:

To the client application, the alias name represents the IMS system, or datastore, to which the requesting application program connects. IMS Connect keeps track of which alias names are managed by which ODBMs in an internal tracking table populated during registration with each instance of ODBM. An alias name can be associated with a single datastore, or it can reference multiple datastores. If a datastore in the ODBM configuration does not have an alias associated with it, then ODBM generates an internal alias name equal to the value of the datastore name.

Depending on the value of the alias name submitted, IMS Connect either routes the incoming connection request to a specific ODBM instance or uses a round-robin distribution of the incoming connection request to any available instance of ODBM in the IMSplex that has defined that alias name. ODBM then routes the database connection requests received from IMS Connect to the IMS systems that own the requested database. If multiple datastores in an ODBM are affiliated with the same alias then a round-robin technique is also used to send the request to a datastore.

ODBM Execution Parameters

- Execution Only Parameters
  - BPECFG=
    - Specifies the BPE configuration parameters PROCLIB member
    - Optional parameter
  - BPEINIT=CSLDINI0
    - Specifies the ODBM start up module required by BPEINI00
    - Must be CSLDINI0 for ODBM address space start up.
    - Required parameter
  - ODBMINIT= 000 | xxx
    - Specifies a 3-character suffix for the ODBM initialization parameters PROCLIB member, CSLDIxxx.
    - Default is 000

105

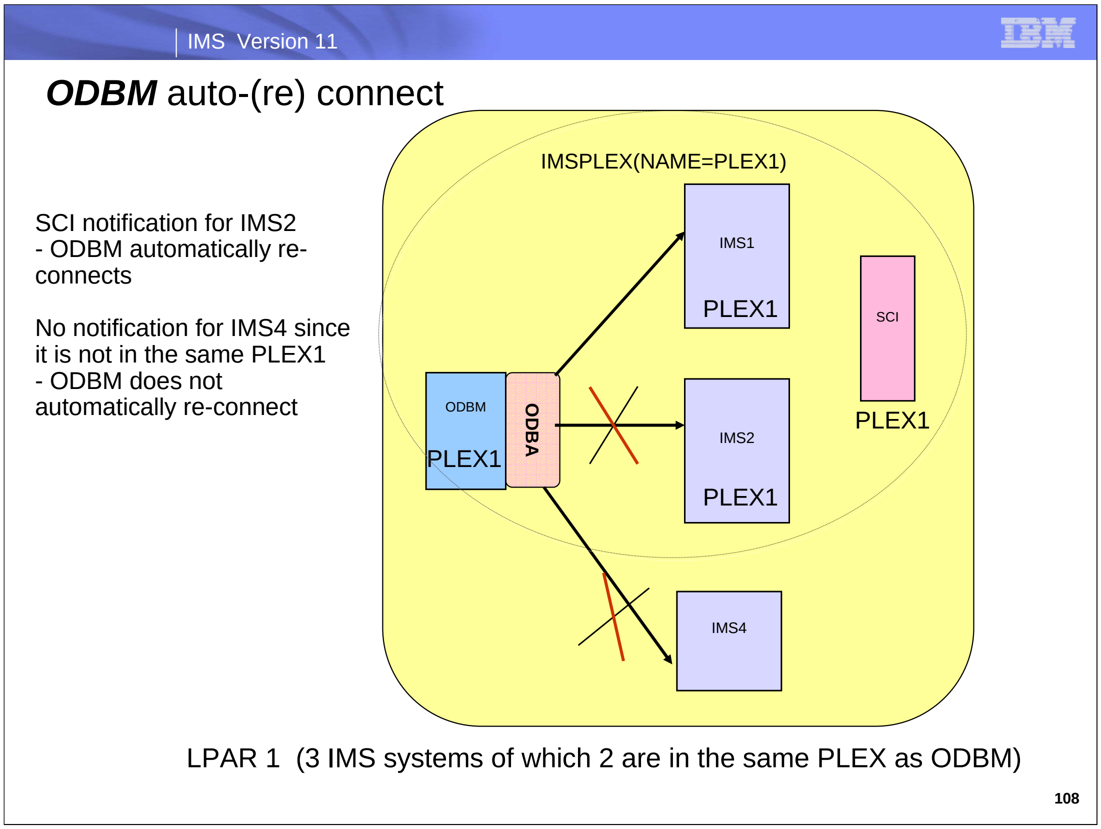
If BPECFG= is not specified, BPE uses default values of: LANG=ENU, TRCLEV=ERR, No user exits.

ODBM Execution Parameters (cont.)

- Execution Optional Parameters
  - override CSLDIxxx parameters
    - ARMRST=Y|N
      - Specifies MVS Automatic Restart Manager (ARM) used to restart the ODBM
    - RRS= Y | N
      - Y - Specifies ODBM registers with RRMS and use the ODBA/DRA interface
      - N - Specifies ODBM is to use the CCTL/DRA interface
    - ODBMNAME
      - Specifies the name for the ODBM address space.
      - Must supply a unique 1-6 character name
      - ODBMID is the ODBMNAME followed by the characters "OD".
        - Used as ODBM Member name for IMSplex Group
    - ODBMCFG=000 | xxx
      - Suffix for the ODBM configuration PROCLIB member, CSLDCxxx

ODBM startup and shutdown

- ODBM can start before or after starting IMS, IMS Connect, RRS
  - S odbm
- STOP CSLJOB command
  - STOP (P) odbmjobname
- CSL SHUTDOWN command
  - f,sciname, SHUTDOWN …
- CSLZSHUT API
  - Programming interface



ODBM automatically connects to IMS systems specified in CSLDCxxx during initialization.

ODBM will only auto-(re)connect with an IMS datastore that has been configured/defined to that ODBM. ODBM attempts auto-(re)connect with the IMS after receiving the SCI notify that this IMS has (re)joined the IMSplex.

Note since IMS does not have to be member of an IMSplex, for these IMS's, SCI will not be able to notify ODBM and ODBM will not be able to auto-reconnect to that IMS.

ODBM BPE Configuration PROCLIB Member

- BPECFG= specifies PROCLIB member for BPE configuration BPE.
  - TRCLEV=(type, level, component [,PAGES=num_pages])
    - Valid ODBM trace table types: CSL, ERR, ODBM, PLEX
  - EXITMBR=(membername, component)
    - Specifies the user exit list PROCLIB member name
    - Valid component specifications are: BPE, ODBM

109

Note this is not the complete list of BPE configuration parameters, only the ones used to identify ODBM information.

TRCLEV specifies the trace level for a trace table and, optionally, the number of pages of storage allocated for the trace table. Valid levels are: NONE, ERROR, LOW, MEDIUM, HIGH

ODBM – Optional User Exits

- CLNTCONN User Exit
  - Called during client command registration and deregistration processing
  - Can be used to monitor ODBM Client access to ODBM
- Initialization/Termination User Exit
  - Called during ODBM and IMSplex initialization and termination
  - Can be used to monitor ODBM and IMSplex init/term processing
- Input User Exit
  - Called to view DL/I calls that are issued to IMS databases
  - Modify or reject the command before it is processed
- Output User Exit
  - Called to view output from ODBM
  - Modify the output before it is returned to the originator of the command

110

Open Database Manager (ODBM) user exits can be written to customize and monitor the ODBM environment.

No sample exits are provided.

CLNTCONN

An exit routine that is called during client command registration and deregistration processing.

INITTERM

An exit routine that is called during various phases of initialization and termination.

INPUT

An exit routine that is called to view DL/I calls that are issued to IMS databases. This exit routine can either modify the command before execution or reject the command before it is processed.

OUTPUT

An exit routine that is called to view output (for example, ODBA call output) from ODBM to an ODBM client. The exit routine can modify the output before it is returned to the originator of the command.

ODBM User Exit List PROCLIB Member Example

```
*******************************************************************
* ODBM USER EXIT LIST PROCLIB MEMBER                            *
*******************************************************************

#---------------------------------------------------------------#
# DEFINE 1 ODBM INIT/TERM USER EXIT: ZDINTM00                   #
#---------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZDINTM00),COMP=ODBM)

#---------------------------------------------------------------#
# DEFINE 1 ODBM INPUT USER EXIT: ZINPUT00                       #
# WITH AN ABEND LIMIT OF 8.                                     #
#---------------------------------------------------------------#
EXITDEF(TYPE=INPUT,EXITS=(ZINPUT00),ABLIM=8,COMP=ODBM)

#---------------------------------------------------------------#
# DEFINE 1 ODBM OUTPUT USER EXIT: ZOUTPUT0                      #
#---------------------------------------------------------------#
EXITDEF(TYPE=OUTPUT,EXITS=(ZOUTPUT0),COMP=ODBM)
```

ODBM – Security

- ODBM does not perform any user authentication or authorization
  - Assumes the end client userid associated with the request has been authenticated and passed by the ODBM client, e.g., IMS Connect, otherwise the userid associated with the ODBM address space will be used
- For ODBA-DRA (RRS=Y)
  - ODBASE= Y for APSB security (AIMS | Axxxxxxx resource class), or
  - ISIS= C|R|A and ODBASE=N for RAS security (IIMS | Ixxxxxxx resource class)
- For CCTL-DRA (RRS=N)
  - ISIS= C|R|A and ODBASE=N for RAS security (IIMS | Ixxxxxxx resource class)

112

IMS provides several options for establishing and defining security for application programs that use the ODBA interface. The options that you select depend on the type of security environment and authorization method that you plan to use. In general, the process that IMS uses to secure PSBs involves one of the following types of security checking:

- APSB security. A security check is performed to determine if the user is authorized to use the PSB
- Resource access security (RAS). A security check is performed by RACF® to determine if the user is authorized to use the PSB. RACF determines authorization by looking at the RACF security class profile defined for the dependent region.

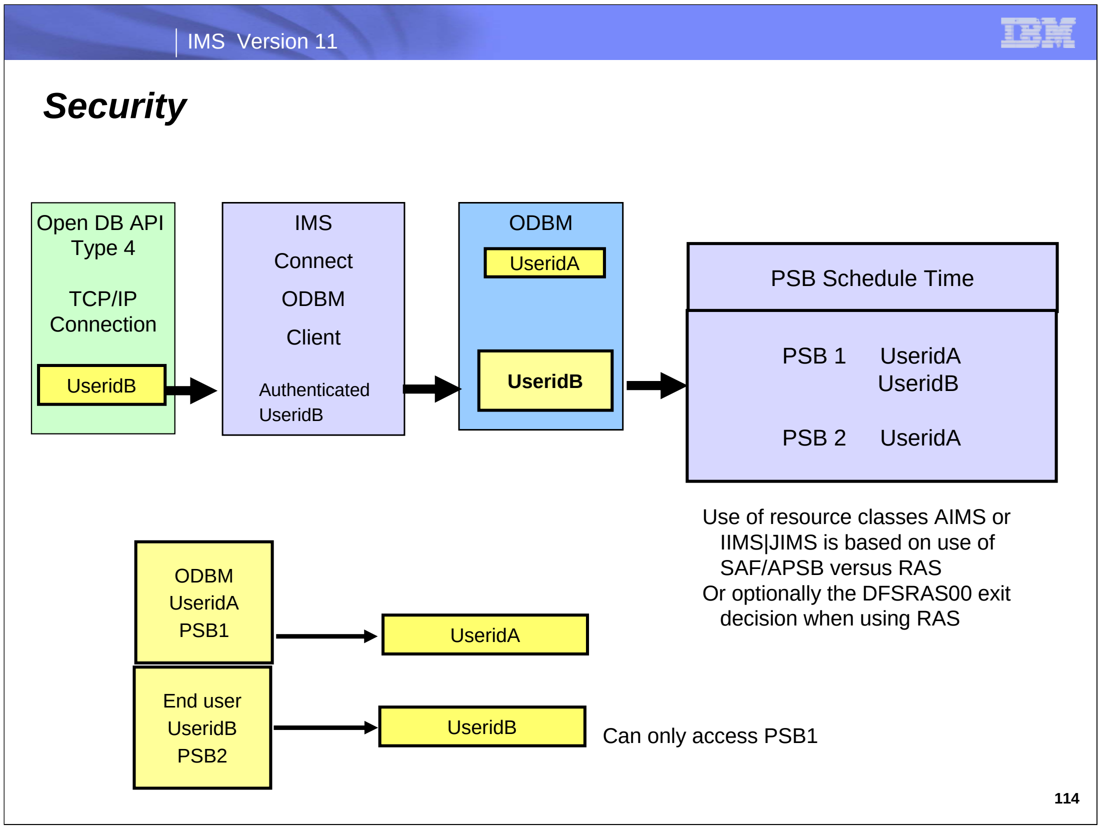
ODBASE= and ISIS= are IMS parameters in the IMS or DBC startup procedure or the DFSPBxxx PROCLIB member.

Security Options

Security implementation	Authorization	ISIS= specification	ODBASE= specification	PSB Security
APSB Security	RACF	Not applicable	Y	X (AIMS | Axxxxxxx)
Resource access security	RACF	R	N	X (IIMS | Ixxxxxxx) (JIMS | Jxxxxxxx)
	Exit – DFSRAS00	C	N	X decided by exit
	RACF and Exit DFSRAS00	A	N	X (IIMS | Ixxxxxxx) (JIMS | Jxxxxxxx) or Decided by exit
	None	0 | N	N	

113

This table identifies the values that you need to specify to control data access for specific security implementations. The table also indicates the type of security checking that is performed for each set of specifications.



A security check is performed to determine if the user is authorized to use the PSB.

ODBM will use the calling userid security environment (ACEE) passed to it from the calling address space, e.g., IMS Connect. If the ACEE is not present then ODBM will use its own userid.

ODBM Type-2 Commands

- QUERY ODBM
  - monitor connections, status, and configuration of ODBM

QRY ODBM TYPE(ALIAS)
QRY ODBM TYPE(CONFIG)
QRY ODBM TYPE(DATASTORE)
QRY ODBM TYPE(SCIMEMBER)
QRY ODBM TYPE(THREAD)
QRY ODBM TYPE(TRACE)

- UPDATE ODBM
  - update connections, status and configuration of ODBM

UPD ODBM START(TRACE)
UPD ODBM STOP(TRACE)
UPD ODBM START(CONNECTION)
UPD ODBM STOP(CONNECTION)
UPD ODBM TYPE(CONFIG)

115

To process commands from an OM, ODBM as a command processing client registers its commands with an OM Commands that are issued to OM are, by default, routed to all ODBMs in the IMSplex that are active and registered to process that particular command.

ODBM Type-2 Commands …

- UPD ODBM START(CONNECTION) DATASTORE( ) or ALIAS( )
- Starts connections to data stores and aliases
  - ODBM Member Name
  - Datastore Name
  - ALIAS Name
- UPD ODBM STOP(CONNECTION) DATASTORE( ) or ALIAS( )
- Stops connections to data stores and aliases
  - ODBM Member Name
  - Datastore Name
  - ALIAS Name

IMS /DIS A Thread
/STOP Thread region number

116

ALIAS() and DATASTORE() are mutually exclusive and there is no default value.   Specifying DATASTORE will start/stop connections for all Aliases associated with the DATASTORE.

When a request to stop existing thread activity is issued, ODBM first waits until the client application program has finished using the database connection. Hung threads are timed out. An IMS /DIS A Thread can also be used to find the thread region number so that the /STOP Thread region number command can be used to stop the specific thread. Note that MVS cancel commands are rejected by IMS.

After an UPDATE ODBM STOP(CONNECTION) to a DATASTORE has been issued, a subsequent UPDATE ODBM START(CONNECTION) DATASTORE() command must be issued in order to restart the connection(s) to the datastore(s). If the actual datastore subsystem(s) are recycled after this command, ODBM will not attempt to connect to the datastore(s) until an UPDATE ODBM START(CONNECTION) DATASTORE() command is issued

ODBM Type-2 Commands …

UPD ODBM TYPE(CONFIG) MEMBER(…) OPTION(..)

- For loading an updated copy of the current or an alternate CSLDCxxx
- All connections must be stopped
  - MEMBER
    - 1-3 xxx value for CSLDCxxx PROCLIB Member to load
    - Default is the current CSLDCxxx PROCLIB member.
  - OPTION
    - CONNECT
      - Default action
      - connections will be made to datastores
      - aliases associated with the data stores are made available
    - NOCONNECT
      - No connections will be attempted

117

Note:

If a new CSLDCxxx member is used to update the active ODBM configuration, the new member becomes the current ODBM configuration PROCLIB member. When restarting ODBM, the ODBMCFG= parameter on the EXEC statement or within the CSLDIxxx initialization PROCLIB member will be used to configure ODBM. All connections must be stopped on an ODBM that processes the UPD ODBM TYPE(CONFIG) command prior to issuing the command.

- OPTION -CONNECT means all connections will be made to the datastores in the updated members.
- OPTION -NOCONNECT Specifies that after updating the active ODBM configuration, no connections will be attempted.

ODBM – Messages , SMF and IMS Log

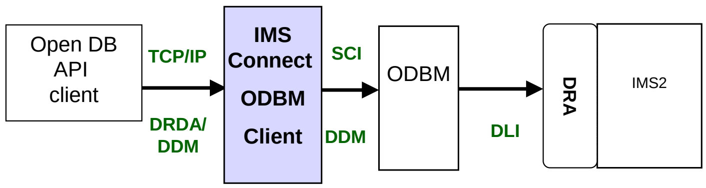
- **ODBM messages begin with CSL4**
- **ODBM registers with SMF to create the SMF 89 records**
- **ODBM adds 16 byte APSB thread token to IMS X'08' log record**

Open Database Manager

- Benefits
  - Allow IMS databases to be processed as a standards-based data server
  - Enables new application design frameworks and patterns
  - Opens access to online IMS database resources
    - Across LPAR boundaries
    - Across Network boundaries
  - Protects IMS control regions from the unexpected termination of the application program
    - Reduce U113 abends

IMS Open Database support enhances the distributed access to IMS data and the availability of IMS.

The IMS Connect ODBM Client



120

Open Database - IMS Connect topics

- DRDA concepts
- IMS Connect configuration member update
- Routing and security
- Enhanced IMS Connect Type-1 commands
- Enhanced z/OS MODIFY interface commands
- New IMS Connect messages
- IMS Connect event records

Open Database - IMS Connect topics

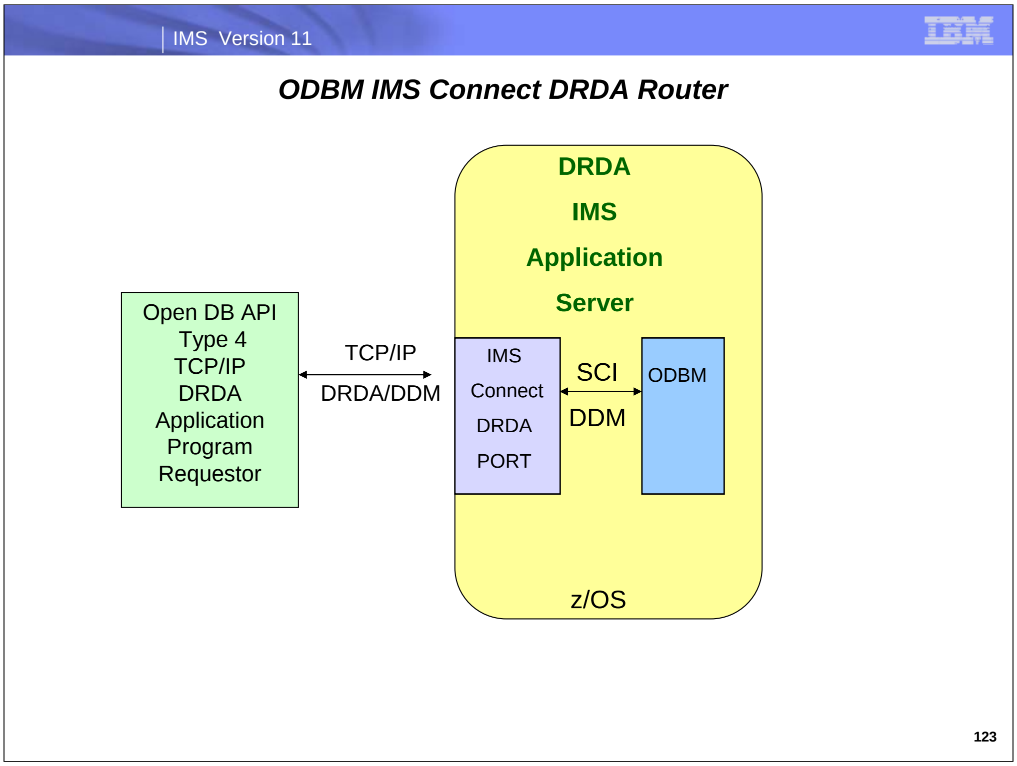
- Distributed Relational Database Architecture (DRDA)
  - Set of protocols and functions for client and database servers connectivity
    - Communication protocol
    - Two-Phase commit protocol
    - Security
  - Distributed Data Management (DDM)
    - Provides the command and reply structure for accessing databases
    - ODBM translates the DDM database requests into DL/I calls
  - IMS Connect is the server for the DDM commands sent via TCP/IP

Distributed Relational Database Architecture (DRDA) is a set of protocols and functions that provide connectivity between a client and database server.

Distributed Data Management (DDM) which is part of DRDA provides the command and reply structure for accessing distributed databases.

IMS Connect is the TCP/IP server for all DDM commands and data that is sent by way of TCP/IP sockets.

ODBM translates the incoming database requests from the DDM protocol submitted by the IMS-provided connectors and user-written DRDA applications into the DL/I calls expected by IMS. ODBM routes the database connection requests received from IMS Connect to the IMS systems that are managing the requested database



ODBM and IMS Connect together form a complete DRDA Application Server.

IMS Connect HWSCFGxx Configuration Member

- New ODACESS statement
  - Defines the communication between ODBM, IMS Connect and remote clients

```
ODACCESS=(ODBMAUTOCONN=Y|N, ODBMTMOT=, DRDAPORT=(ID=,KEEPAV=,PORTTMOT=),
          IMSPLEX=(MEMBER=,TMEMBER=))
```

- ODBMAUTOCONN=Y|N specifies whether or not IMS Connect is to register with all current and future ODBMs that enter the IMSplex
- ODBMTMOT= defines the time in hundredths of a second IMS Connect waits for both:
  - A response message on Database connections with ODBM
  - An initial input message after a socket connection is established with a client
  - 0 disables the timeout function, default is 18000 (3 minutes)
- DRDAPORT=
  - ID=1- to 5-numeric character decimal field to define DRDA port.
  - KEEPAV= 1- to 8-character decimal field sets TCP/IP keep alive mechanism
  - PORTTMOT= amount of time IMS Connect waits for the next input message from a client application decimal integer in hundredths of a second
- IMSPLEX=(MEMBER=,TMEMBER=)
  - MEMBER= specifies the IMS Connect member name used to join the IMSPLEX group.
  - TMEMBER= specifies the IMSPLEX group IMS Connect joins.

124

- ODACCESS - Defines characteristics of the communication between ODBM, TCP/IP DRDA clients and IMS Connect. Note only DRDAPORT= is needed to define IMS Connect as a DRDA server.
- ODBMAUTOCONN specifies whether or not IMS Connect automatically connects to new and existing instances of ODBM within an IMSplex. If set to N and ODBM is started IMS Connect will not register with that ODBM. A STARTOD command is required.
- ODBMTMOT specifies for IMS Connect connections with ODBM the amount of time in hundredths of a second IMS Connect waits for: a response message on connections with ODBM, or an initial input message after a socket connection is established on connections with a client application. Specifying a timeout value can avoid hang conditions when an ODBM instance stops responding. For connections with a DRDA TCP/IP 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.
- DRDAPORT= defines the port numbers, the TCP/IP keepalive value, and the timeout values for the DRDA Clients.
  - ID=1-5 DRDA PORT number. This parameter supports a max of 50 ports combined between DRDAPORT, PORTID, PORT, and SSLPORT. The specified value must not conflict with any other ports selected in the TCP/IP domain and in IMS Connect configuration.
  - KEEPAV sets the override interval for the z/OS TCP/IP layer keepalive mechanism.
  - 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.

Note For client connections, ODBMTMOT differs from PORTTMOT in that ODBMTMOT applies only to the first input message after a socket connection is established and PORTTMOT applies only to input messages that follow a previous input message.

- IMSPLEX() specifies IMSPLEX that IMS Connect communicates with through the IMS Structure Call Interface (SCI). This is not a required field. IMS Connect will use the global IMSPLEX statement. If that is not defined, IMS Connect will still start but will not be able to access ODBM and will reject any request for ODBM processing.

IMS Connect HWSCFGxx Configuration Member . . .

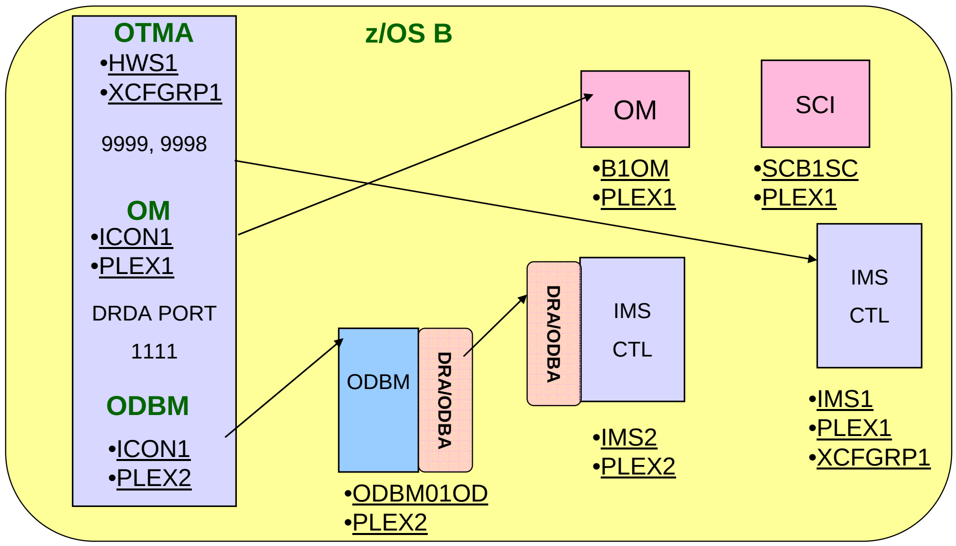
- Example of ODACCESS statement

```
HWS=(ID=HWS1,XIBAREA=100,RACF=N)
   TCPIP=(HOSTNAME=TCPIP,PORTID=(9999,9998,LOCAL),
        EXIT=(HWSSMPL1,HWSCSLO1,HWSSOAP1))

   DATASTORE=(ID=IMS1,GROUP=XCFGRP1,MEMBER=HWS1,
           TMEMBER=IMS1,DRU=HWSYDRU0,
           APPL=APPLID1,RRNAME=RTPIPE)

   IMSPLEX=(MEMBER=ICON1,TMEMBER=PLEX1)

   ADAPTER=(XML=Y)

   ODACCESS=(ODBMAUTOCONN=Y,ODBMTMOT=6000,
       DRDAPORT=(ID=1111,KEEPAV=5,PORTTMOT=50),
         IMSPLEX=(MEMBER=ICON1,TMEMBER=PLEX2))
```

125

In this example, there is one DRDA port, ODBM auto connect option set to yes, timeout of 60 seconds, and a dedicated IMSplex for ODBM messages. All IMS commands for OM pass through the general IMSPLEX specification. All DRDA messages for ODBM will go through the ODACCESS IMSPLEX.

IMSPLEX = CSLPLEX1 and CSLPLEX2



IMS Connect can be configured to join the IMS OTMA XCF group in PLEX1,
the IMSPLEX group for OM communications in PLEX1 and the IMSPLEX group for ODBM in PLEX2.

126

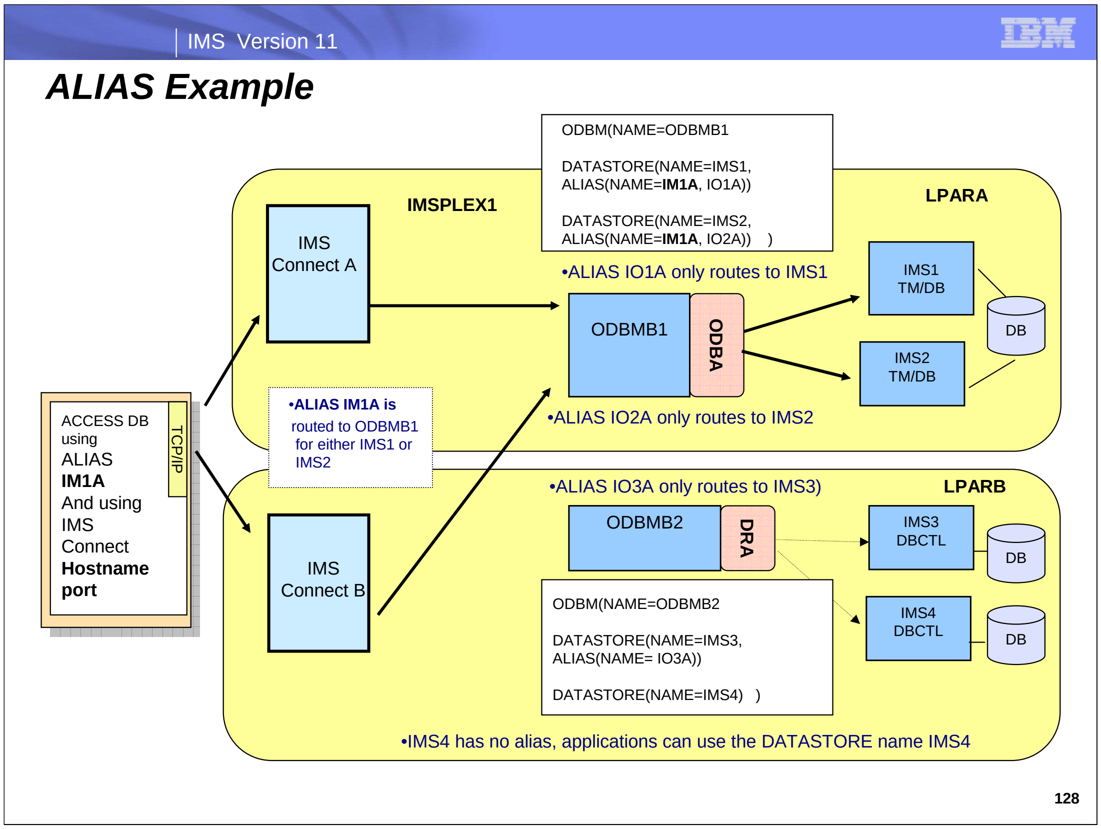
This is an example of an IMSPLEX in a single z/OS LPAR. IMS Connect can be configured to join the IMS OTMA XCF group in PLEX1, the IMSPLEX group for OM communications in PLEX1 and the IMSPLEX group for ODBM in PLEX2.

IMS Connect Routing to ODBM

- ODBM Alias Name routing
  - Alias name used by IMS Connect DRDA Client application programs
    - Do not need to know IMSID
  - IMS Connect round robin routing
    - Blank alias name
    - Alias name managed by multiple ODBMs

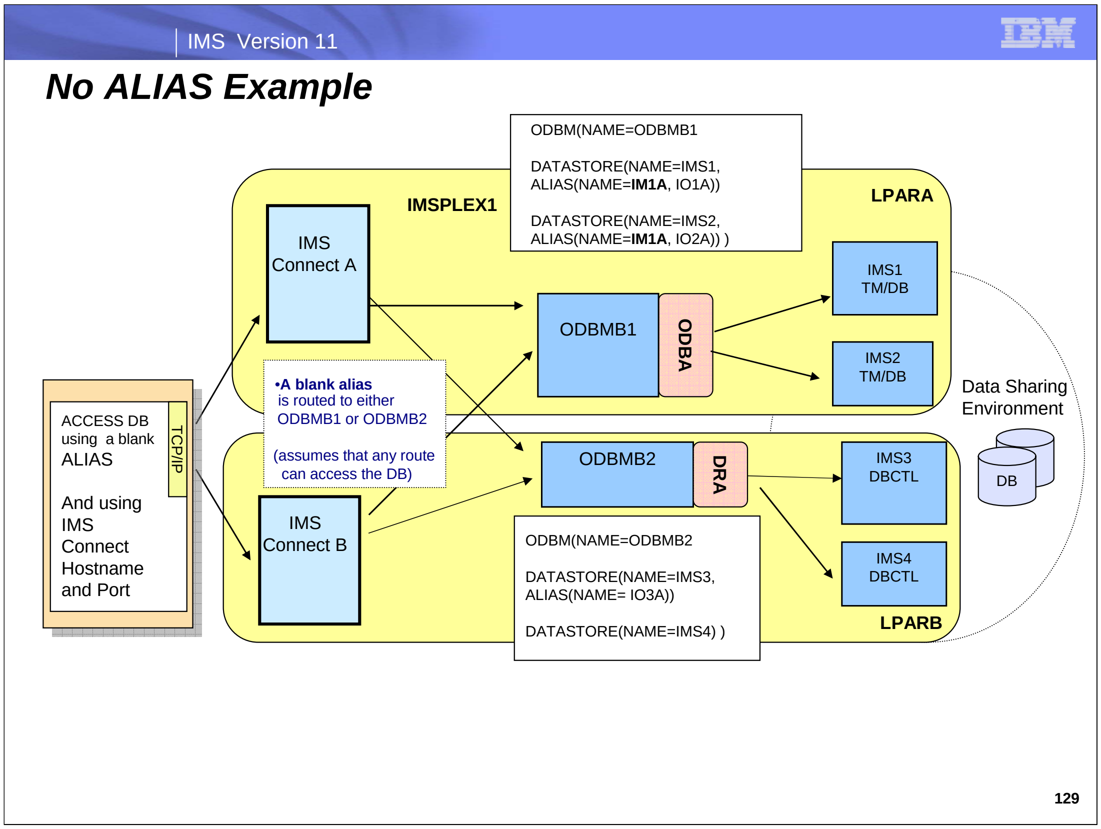
127

IMS Connect routes incoming connections from client applications based on an alias name submitted by the client application and the instance of ODBM to which the alias name belongs. IMS Connect keeps track of which alias names belong to which instance of ODBM in an internal tracking table populated during registration with each instance of ODBM. If the alias name specified by the client application is owned by more than one instance of ODBM or if the client application leaves the alias name blank, IMS Connect routes incoming connections in round-robin fashion among the instances of ODBM within the IMSplex that own that alias name.



This is an example of an IMSplex. Application programs can use the alias name to access IMS data stores without needing to know the actual IMSID of the IMS data store.

In this example, the remote program requests access to the IMS database environment associated with alias IM1A. Because IM1A is defined as an alias in ODBMB1, the request is sent to that instance of ODBM which can choose either IMS1 or IMS2 as the target datastore. Data sharing has been implemented and either IMS can process the request. On the other hand, a request that uses alias IO1A can only be sent to IMS1, one that specifies IO2A will be sent to IMS2, and one that specifies 1O3A will only be sent to IMS3. Although IMS4 does not have an alias, an internal alias name of IMS4 is created that supports requests using that name and associated requests can only be routed to ODBMB2 because that is the only ODBM address space in this configuration that is connected to IMS4



If the client application leaves the alias name blank, as shown in this example, IMS Connect routes incoming connections in round-robin fashion among all the instances of ODBM within the IMSplex and across the IMS systems within those ODBMs.

Regardless of which technique is used, therefore, balancing the database requests in this environment to more than one target assumes that data sharing has been implemented across the alias targets or, for blank aliases, across the IMSplex. Otherwise, the request might be routed invalidly to an IMS that does not have access to the required data.

HWSROUT0

- IMS Connect DB Routing Exit routine (HWSROUT0)
  - Can determine or change the destination of a database access request. The following choices are examples of IMS Connect actions based on how the exit routine is coded:
    - If HWSROUT0 selects an ODBM instance by its ODBM name, IMS Connect immediately honors the request.
    - If HWSROUT0 does not select a specific ODBM but rather specifies an alias name or leaves it blank then IMS Connect selects an ODBM instance.
      - If an alias name is chosen and only one ODBM has specified the name, then the request is routed to that ODBM instance.
        - If the alias is associated with multiple ODBMs then IMS Connect uses a round-robin technique to route the request across those resources.
      - If the alias is blanks then IMS Connect round-robins the request across all the ODBM resources in the IMSplex.
    - If HWSROUT0 overrides the IMS alias provided in the incoming request, IMS Connect takes the appropriate actions for alias names

130

As with the other IMS Connect exit routines in the transaction environment, this new DB Routing Exit Routine provides the interface for a creative programmer to code a solution that answers both a failover as well as load balancing need for database requests. Alternatively, products such as *IMS Connect Extensions (5655-S56)* automate these same actions through configuration specifications without the need for the exit to be user coded.

HWSAUTH0

- IMS Connect DB Security User Exit (HWSAUTH0)
  - Can perform authentication of user IDs or RACF PassTickets for IMS DB clients.
  - Can override the input user ID with a different user ID
  - Can provide a RACF group ID to be authenticated further by IMS Connect.

131

You can use the IMS Connect DB security user exit routine (HWSAUTH0) to authenticate the input user ID and password or passticket specified by IMS Connect clients that access IMS DB, such as any of the IMS Universal drivers or a client application program that connects to IMS Connect through a user-written DRDA source server. HWSAUTH0 can also be used to override the input user ID with a different user ID and to provide the RACF group ID to be authenticated further by IMS Connect. IMS Connect always calls HWSAUTH0 to perform the client authentication independently from the RACF parameter in the IMS Connect configuration member. IMS Connect calls HWSAUTH0 before invoking any installed security facility such as RACF.

IMS Connect always calls security exit HWSAUTH0 to authenticate the user ID and password. New for IMS V11 is that IMS Connect then calls RACF to create a RACO if RACF= Y is specified in the IMS Connect configuration member.

IMS Connect User Exits for ODBM

- BPE managed and refreshable User Exits
  - Routing user exit – HWSROUT0
    - Override the IMS Connect selection of an ODBM
  - Security user exit – HWSAUTH0
    - Authenticate the input user ID and password or passticket
    - Provide the RACF group ID to be authenticated
    - Always called by IMS Connect

```
EXITDEF(TYPE=ODBMROUT,EXITS=(HWSROUT0),ABLIM=8,COMP=HWS)

EXITDEF(TYPE=ODBMAUTH,EXITS=(HWSAUTH0),ABLIM=8,COMP=HWS)
```

132

HWSROUT0 routing exit can be coded by customers to override the IMS Connect selection of an ODBM

HWSAUTHO is the IMS Connect DB security user exit routine and can be used to:

1. authenticate the input user ID and password or passticket
2. override the input user ID with a different user ID and to provide the RACF group ID to be authenticated by IMS Connect

IMS Connect will always perform the security checking of the end client that is requesting ODBM services via the new security exit HWSAUTH0, independently from the RACF= parameter in the IMS Connect configuration member.

ODBM BPE Trace Tables

- CSL
  - Used for routines that are common to all CSL managers
- ERR
  - Used to trace errors that occur within the ODBM address space
- ODBM
  - Used for general ODBM processing flow
- PLEX
  - Used for ODBM processing for a specific IMSplex

BPE Commands

```
F,ODBM, DISPLAY TRACETABLE NAME(*) OWNER(ODBM)
F,ODBM,UPDATE TRACETABLE NAME(*) OWNER(ODBM) LEVEL(HIGH) EXTERNAL(YES)
```

133

ODBM trace table types

CSL

The Common Service Layer (CSL) trace table is used for routines that are common to all CSL managers.

ERR

This table is used to trace errors that occur within the ODBM address space.

ODBM

This table is used for general ODBM processing.

PLEX

This table is used for ODBM processing for a specific IMSplex.

BPE z/OS modify commands can be used to display and update ODBM trace tables

ODBM BPE Configuration PROCLIB Member Example

```
TRCLEV=(*,LOW,ODBM)              /* DEFAULT ODBM TRACES TO LOW */
TRCLEV=(CSL,HIGH,ODBM)           /* CSL TRACE ON HIGH          */
TRCLEV=(ODBM,HIGH,ODBM)          /* ODBM GENERAL TRACE ON HIGH */
TRCLEV=(PLEX,HIGH,ODBM)          /* IMSPLEX TRACE ON HIGH      */

#
# USER EXIT LIST PROCLIB MEMBER SPECIFICATION
#

EXITMBR=(CSLEXOB0,BPE)           /* SPECIFY PROCLIB DATASET   */
                                 /* MEMBER CSLEXOB0 AS BPE'S  */
                                 /* USER EXIT LIST MEMBER     */
EXITMBR=(CSLEXDM0,ODBM)          /* SPECIFY PROCLIB DATASET   */
                                 /* MEMBER CSLEXDM0 AS ODBM'S */
                                 /* USER EXIT LIST MEMBER     */
```

134

ODBM User Exit List PROCLIB Member

- EXITMBR= specifies PROCLIB to define ODBM user exits
  - EXITDEF = (TYPE=type, EXITS=(exitname1, exitname2, ...) [,ABLIM=limit], COMP=ODBM)
    - TYPE=type
    - EXITS=(exitname)
      - list of one or more user exit module names.
    - ABLIM=limit
      - limit for the exit type being defined.
      - decimal number from 0 to 2147483647.

F,ODBM,DISPLAY USEREXIT …

F,ODBM,REFRESH USEREXIT …

EXITDEF is used to associate a user exit type with a list of one or more user exit modules to be called. The exits are called in the order listed.

Enhanced IMS Connect Type-1 Commands

- VIEWPORT
  - Provides IMS Connect DRDA port information
- VIEWDS ODBM
  - Display the current status and activity for specified ODBM member
- VIEWHWS SUMMARY
  - Provides IMS Connect ODBM information

VIEWPORT	*ALL|PORTID|LOCAL*
VIEWDS	*ALL|DATASTORE name|ODBM Name|IMSPLEXID*
VIEWHWS	*SUMMARY*

136

Several of the IMS Connect Type-1 Commands have been enhanced to display open database information.

The VIEWPORT command display the status, clients, and activity of a port. The STATUS display shows the state of the client's thread as well as the 16-byte APSB-TOKEN which is managed by ODBM.

The VIEWDS ODBM command shows if IMS Connect is registered with the identified instance of ODBM or if the identified instance of ODBM is not available. The command output also indicates the RRS specification in an ODBM, the alias name(s) associated with the datastores defined to that ODBM, and whether the IMS system associated with the preceding alias name is active or inactive.

The VIEWHWS command displays the current activity of IMS Connect. With the SUMMARY option, the output provides the same HWS and data store information but bypasses listing each individual client for the ports. The port status and client totals are still reported.

New IMS Connect Type-1 Commands

STARTOD and STOPOD	*ODBMNAME*
STARTIA and STOPIA	*ALIASNAME|ODBMNAME*
VIEWIA	*ALIASNAME|ODBMNAME*
SETOAUTO	*YES|NO*

Enhanced z/OS MODIFY IMS Connect Commands

(F** imsconnectname**, QUERY…) includes information about ODBM

QUERY IMSPLEX NAME(imsplexName|*) SHOW(ALL)
QUERY DATASTORE (NAME|*) SHOW(ALL)
QUERY MEMBER TYPE(IMSCON) SHOW(ALL|SUMMARY)
QUERY PORT NAME (NAME|*) SHOW(ALL)
QUERY ALIAS (NAME|*) ODBM(NAME|*)

137

IMS Connect introduces several new Type-1 commands that reflect the new connections to ODBM:

- Use the STOPOD command for any type of error situation that requires immediate termination of communication with an ODBM. Work currently in progress for an ODBM is ended and communications with that ODBM and its threads are terminated. After a STOPOD command, the ODBM status will be UNREGISTERED. Likewise, you can manually start ODBM with the STARTOD command which will set the ODBMstatus to REGISTERED.
- The STARTIA and STOPIA commands work with access to IMS alias names. The STARTIA command sets the specified alias to active so that IMS Connect can route to that alias.
- The STOPIA command sets the specified alias to inactive so that IMS Connect cannot route to that alias name.
- The VIEWIA command displays the status of IMS aliases and their associated ODBMs.
- You can use the SETOAUTO command to specify whether IMS Connect automatically connects to ODBM when either IMS Connect is started or an instance of ODBM is activated in the IMSplex. When automatic connection with ODBM is set to YES, IMS Connect will connect to all the future ODBMs that enter the IMSplex. This option can be specified in the ODBMAUTOCONN= parameter of the IMS Connect configuration member. When automatic connection with ODBM is set to NO, IMS Connect will not register with any future ODBMs that enter the IMSplex. After turning off the automatic connection of IMS Connect to ODBM, you can connect to ODBM manually by using the STARTOD command to open communication with an instance of ODBM.

Additionally, IMS Connect has enhanced the z/OS MODIFY (F *imsconnectname*, QRY…) interface to include information about ODBM.

QUERY IMSPLEX NAME(imsplexName|*) SHOW(ALL) – supports TYPE(ODBM)

Enhanced z/OS MODIFY IMS Connect Commands . . .

- Example

```
F HWS,QRY MEMBER TYPE(IMSCON) SHOW(ALL)
ODBM AUTO CONNECTION=Y
ODBM TIMEOUT=18000
ODBM IMSPLEX MEMBER=PLEX2
PORT=1111D STATUS=ACTIVE SOC=2 KEEPAV=5 TIMEOUT=6000
ODBM=ODBM01 STATUS=REGISTERED ODBMRRS=Y
        ALIAS=I01A STATUS=ACTIVE,TRACE
        ALIAS=I01B STATUS=ACTIVE
        ALIAS=I02A STATUS=ACTIVE
        ALIAS=I02B STATUS=NOT ACTIVE
ODBM=ODBM02 STATUS=NOT ACTIVE
```


Enhanced z/OS MODIFY IMS Connect Commands …

UPDATE MEMBER TYPE(IMSCON) START/STOP(TRACE) SET OAUTO(ON|OFF) PSWDMC(ON|OFF) RACF(ON|OFF) RRS(ON|OFF)
UPDATE ALIAS (NAME|*) ODBM(NAME|*) START/STOP(ROUTE)
UPDATE IMSPLEX NAME(imsplexName|*) START/STOP(COMM)
UPDATE ODBM NAME(NAME|*) START/STOP(COMM)

139

IMS Connect also provides the ability using the MODIFY command interface to update the characteristics of communication with ODBM.

New Event Records for HWSTECL0 Exit

Event nbr	Event key	Event Description
41	EVNT	ODBM registration begins.
42	EVNT	ODBM registration ends.
43	EVNT	ODBM de-registration begins.
44	EVNT	ODBM de-registration ends.
91	SVT Token	DRDA command is received.
92	SVT Token	DRDA command is sent.
93	SVT Token	An APSB command is received.
94	SVT Token	An APSB reply is returned.
95	SVT Token	A DPSB command is received.
96	SVT Token	A DPSB reply is returned.

Event nbr	Event key	Event Description
97	EVNT	ODBM Routing Exit entered.
98	EVNT	ODBM Routing Exit returned.
99	EVNT	ODBM Security Exit entered.
100	EVNT	ODBM Security Exit returned.
101	SVT Token	RRS Parent UR token creation begins.
102	SVT Token	RRS Parent UR token creation ends.
103	SVT Token	RRS Set Work Identifier begins.
104	SVT Token	RRS Set Work Identifier ends.
105	SVT Token	Message is sent to ODBM.
106	SVT Token	Message is received from ODBM.

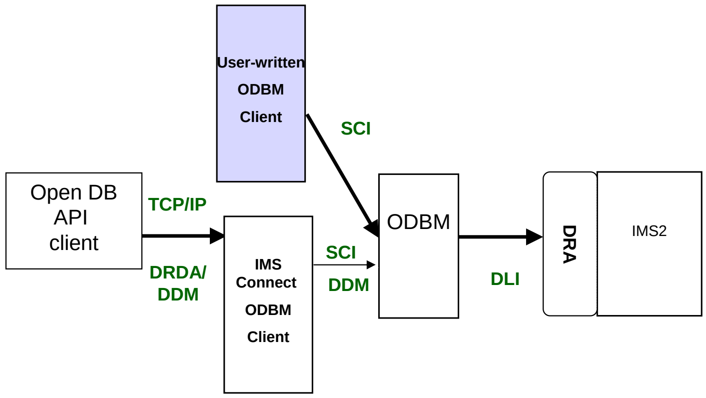
140

This is a list of new event records for this exit. IMS Connect Extensions is one of the exits that will use them.

Open DataBase – IMS Connect

- Benefits
  - Provides TCP/IP access to IMS database resources
  - Supports Open standard client application program

Open Database ODBM Client



142

Open Database Clients

- CSLDMI Interface
  - Provides a set of Assembler Language callas
    - CSLSCREG - Register to SCI
    - CSLDMREG - Register to ODBM
    - CSLSCRDY - Enable the ODBM client for SCI processing
    - CSLDMI FUNC= - API function calls
    - CSLSCBFR – Release output buffer
    - CSLDMDRG – Deregister from ODBM
    - CSLSCDRG – Deregister from SCI
- Benefit
  - Allows user-Written z/OS ODBM assembler language clients to pass DL/I calls to ODBM
    - To access online IMS databases on any LPAR

143

The CSLDMI API provides an interface a Roll Your Own ODBM client can use to access IMS databases on any LPAR. To use the CSLDMI API the ODBM client has to be a z/OS application program written in the assembler programming language.

Interface calls:

CSLSCREG - registers to SCI,enables the client to send ODBM requests to ODBM through SCI.

CSLSCRDY - readies the ODBM client to SCI, which routes messages to the client by client type.

CSLDMREG - registers client to ODBM to enable communication with ODBM.

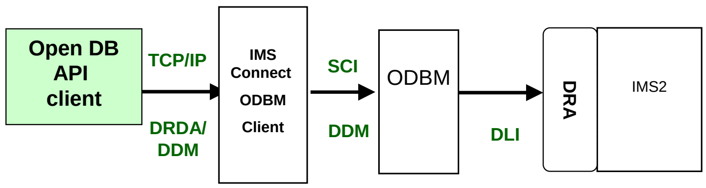
CSLDMI FUNC= IMS DLI calls

CSLSCBFR - releases the output buffer returned by the request.

CSLDMDRG - deregisters client from ODBM to end communications with ODBM.

CSLSCDRG - deregisters from SCI.

Open Database API



Open Database API

- Capabilities
  - Provide access to IMS data directly from z/OS and distributed environments
  - Standards-based approach
    - Java Connector Architecture
    - JDBC, SQL
    - DRDA
- Three Universal drivers for client applications
  - Universal DB resource adapter
    - JCA 1.5
    - Common Client Interface (CCI)
  - Universal JDBC driver
    - JDBC 3.0
  - Universal DLI driver

A “Driver” is the java code that provides an API to a java application, and interfaces to the target DB system

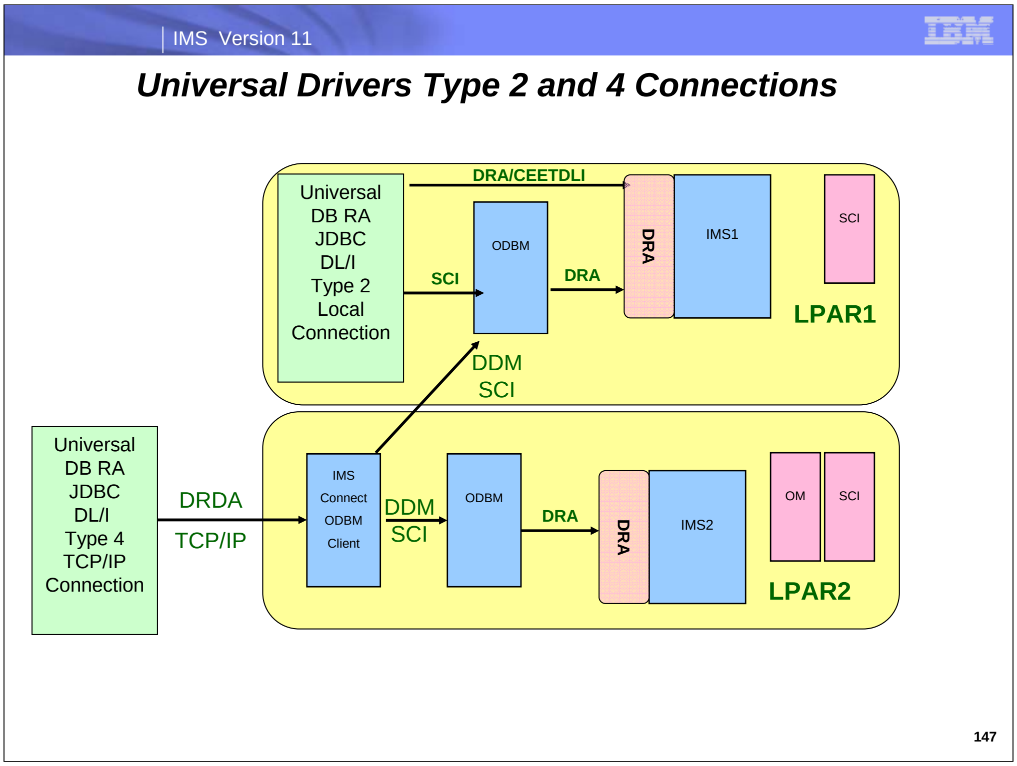
IMS provides a set of drivers that allow you to write Java applications to access IMS databases from z/OS (same or different LPAR from the IMS database) as well as from distributed (non-z/OS) platforms. The three universal drivers include:

- IMS Universal DB resource adapter – which is a JEE Connector Architecture (JCA) 1.5-compliant resource adapter. It also supports common client interface (CCI) API which provides access from Java™ EE clients to backend enterprise information systems (EIS) such as IMS. Applications that follow the CCI programming interface model have a common structure, independent of the EIS that is being used. The JEE Connector Architecture (JCA) specification defines the CCI.
- IMS Universal JDBC driver – which implements the JDBC 3.0 Java API specifications.
- IMS Universal DL/I driver – which provides a Java API for making calls with traditional DL/I programming semantics.

Recommendation: Over time, ensure that all your Java applications use the IMS Universal drivers

Open Database API

- Local access – Type 2 connectivity to IMS databases
  - Supports access from applications on the same LPAR as IMS
    - IMS JMP/JBP
    - DB2 SP
    - CICS
    - WAS z/OS
- Distributed access – Type 4 connectivity to IMS databases
  - Supports access from applications on a different LPAR or from distributed TCP/IP-enabled platforms and runtimes
    - Windows XP, Vista
    - zLinux
    - z/OS
    - WebSphere Application Server
    - Standalone Java Standard Edition
  - Resource Recovery Services (RRS) is NOT required if applications do not use distributed two phase commit



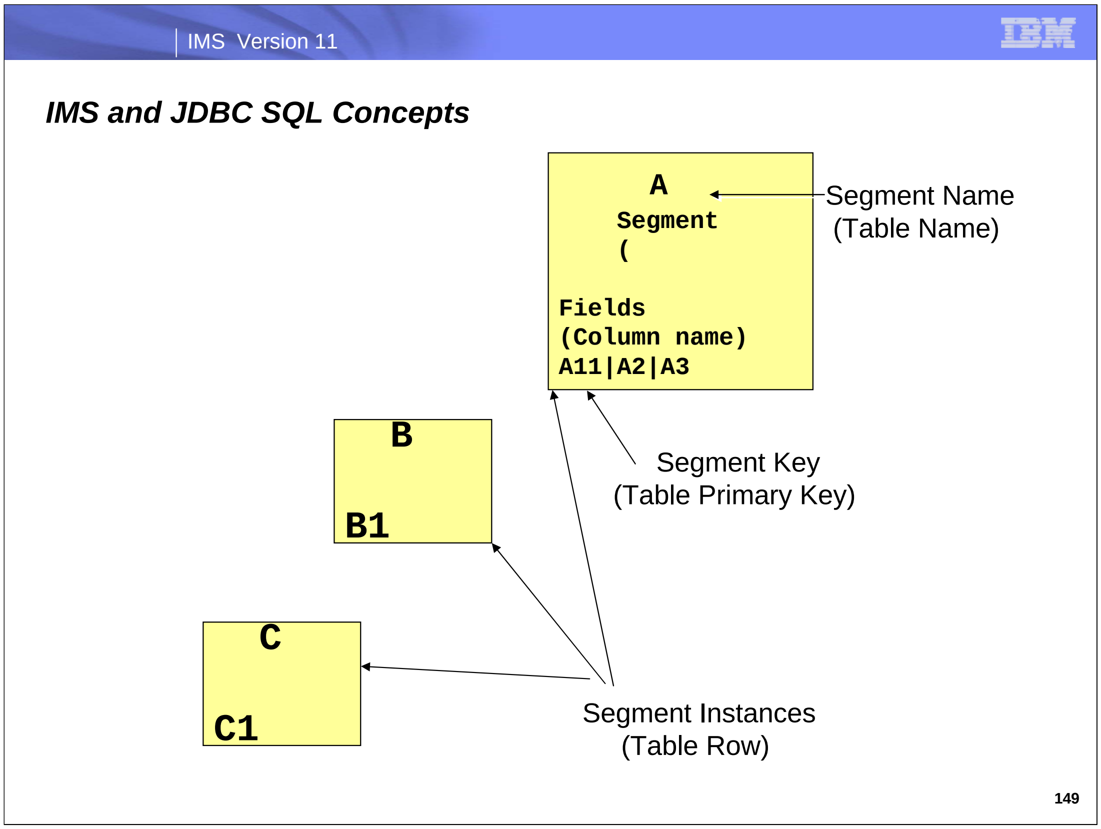
This slide provides an overview of the capabilities provided by Open Database API Universal Drivers.

The Open Database API Type 2 drivers support connections for Local(same LPAR as IMS) access to IMS. When used in IMS JMP/JBP regions the CEETDLI interface is used. For WAS z/OS , DB2 SP and CICS the DRA interface is used.

The Type 4 drivers support distributed access to IMS. Note distributed access includes connections across a network or across an LPAR boundary. DRDA is the protocol used to communicate to IMS Connect. DRDA is the industry standard for DB access in a distributed transaction processing environment. Any required Two Phase Commit and Security flows are imbedded in the DRDA protocol. DRDA uses the distributed data management (DDM) architecture which defines the syntax and semantics of all commands that are sent from a requester to a database server and all reply messages that are sent from a database server to a requester. IMS uses a subset of the existing DDM-defined commands, parameters, and messages, as well as a variety of IMS-defined structures that conform to the DDM architecture, but that are unique to IMS. As implemented by IMS, the DDM commands provide all of the necessary commands to connect to IMS and access IMS databases. IMS DDM commands are used to establish connections, execute IMS DLI calls, commit and rollback unit of work.

Open Database API

- Software requirements include one or more of the following conditional requirements:
  - Java Development Kit (JDK) 5.0 or later
  - For CICS applications, CICS Transaction Server for z/OS Version 3.1 or later
  - For DB2 stored procedures:
    - DB2® for z/OS® Version 9.1
    - DB2 UDB for z/OS Version 8 or later
      - Access from JMP or JBP regions apply APAR PQ74629 to the DB2 subsystem.
  - For Websphere applications
    - WebSphere® Application Server for z/OS Version 6.1 or later
    - WebSphere Application Server for distributed platforms Version 6.1 or later
- IMS JMP and JBP regions require Java Development Kit (JDK) 6.0 or later



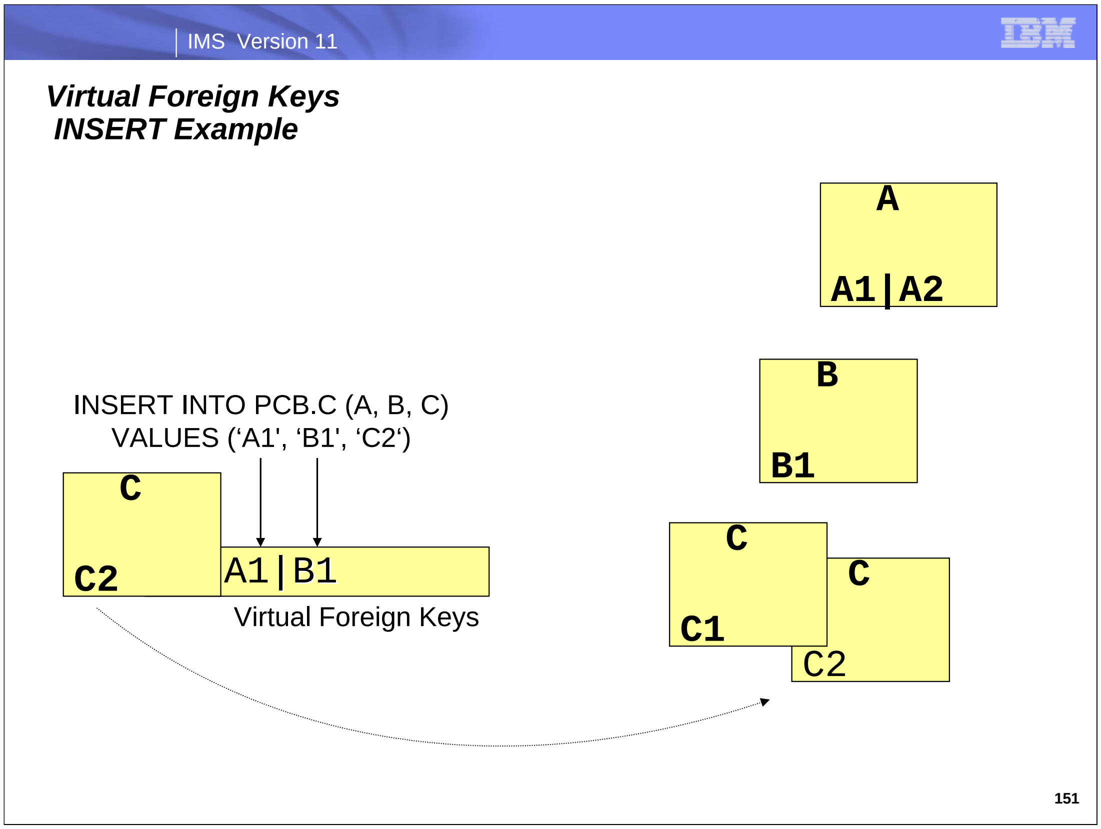
This slide provides a mapping of IMS hierarchical database concepts and relational database concepts.

IMS support for JDBC lets you write Java applications that can issue dynamic SQL calls to access IMS data and process the result set that is returned in tabular format. The IMS drivers support a subset of the SQL syntax with functionality that is limited to what the IMS database can do natively.

JDBC is an application programming interface (API) that Java applications use to access relational databases or tabular data sources

Open Database API JDBC enhancements

- Virtual Foreign Key fields
  - IMS Java maintains the unique keys for segments up to the root
  - SQL SELECT, INSERT, UPDATE, and DELETE queries
    - SQL syntax for IMS appears similar to standard SQL
- Updatable Result Sets
  - Update or delete of current row (IMS segment)
- Metadata Discovery
  - Access the IMS Java Metadata classes generated by the DLIModel utility
- autoCommit support
  - updates are committed as they happen
- setFetchSize
  - An application can set the expected or desired number of rows (IMS segment occurrences) to be returned



In relational databases, hierarchies can be logically built by creating foreign key relationships between tables. In IMS, the hierarchies are explicit and are part of the database definition itself.

The IMS Open Database API introduces the concept of virtual foreign keys to capture these explicit hierarchies in a relational sense, which makes the SQL syntax for IMS appear similar to standard SQL.

When accessing IMS databases every segment that is not the root segment in a hierarchic path will virtually contain the unique keys of all of its parent segments up to the root of the database. These keys are called virtual foreign key fields. The purpose of the virtual foreign key fields is to maintain referential integrity, similar to foreign keys in relational databases. This allows SQL SELECT, INSERT, UPDATE, and DELETE queries to be written against specific tables and columns located in the IMS database hierarchic path.

Remember: Virtual foreign keys are maintained internally and are not physically stored in the IMS database. Virtual Foreign Keys are the same concept as IMS fully concatenated keys. Virtual foreign keys are maintained internally by the IMS Universal drivers.

Updatable Result Sets Example

Query IMS Database

A

A1|A2

```
rs = st.executeQuery("SELECT A2 FROM PCB.A");

while(rs.next()){
            rs.updateString("A2", "A22");
            rs.updateRow();
                }
```

While processing result set update IMS Database

A

A1|A22

autoCommit occurs when the result set is closed or has no more rows

152

A ResultSet object is a table of data representing a database result set, which is usually generated by executing a statement that queries the database. In this example the A2 field in segment A is updated to A22.

Autocommit occurs when the result set is closed or has no more rows.

Open Database API

- Packaging
  - **imsudb.jar**
    - Contains IMS Universal JDBC driver and IMS Universal DL/I driver
  - **imsudbXA.rar**
    - Contains the two-phase commit capable IMS Universal DB resource adapter
  - **imsubLocal.rar**
    - Contains the local transaction capable IMS Universal DB resource adapter

Universal Drivers

IMS Universal DB Resource Adapter

- For accessing IMS data from a JEE application server
  - Supports JEE Connector Architecture 1.5
    - Supports JDBC and Common Client Interface (CCI)
  - Supports both Type 2 and Type 4 connections
  - Provides LocalTransaction support
    - Single-phase commit
  - Provides XATransaction support
    - Global two-phase commit
- Benefit
  - Exploits JCA 1.5 in any JEE Java environment (e.g. in WAS), and supports JDBC SQL calls

155

IMS provides the IMS Universal DB Resource Adapter for accessing IMS data from a JEE application server. JCA is the Java standard for connecting Enterprise Information Systems (EISs) such as IMS into the Java EE framework. Using JCA, you can simplify application development and take advantage of the services that can be provided by a Java EE application server, such as connection management, transaction management, and security management. The Common Client Interface (CCI) is the interface in JCA that provides access from Java EE clients, such as Enterprise JavaBean (EJB) applications, JavaServer Pages (JSP), and Java servlets, to backend IMS subsystems. JDBC is the SQL-based standard interface for database access. It is the industry standard for database-independent connectivity between the Java programming language and any database that has implemented the JDBC interface

IMS Universal JDBC Driver

- Provides a stand-alone JDBC 3.0 driver for making SQL-based database calls to IMS databases
  - Supports JDBC 3.0 SQL-based database connectivity
    - JDBC API
    - Dynamic SQL calls
  - Supports Type 2 and Type 4 connections
  - Provides LocalTransaction support
  - Provides XATransaction support

- Benefit
    - Supports JDBC SQL calls to IMS databases in any stand-alone Java environment

156

For accessing IMS data a Java application (outside a JEE application server) use the IMS Universal JDBC Driver.

IMS Universal DL/I Driver

- Provides a stand-alone Java API for writing granular queries to IMS databases using programming semantics similar to traditional DL/I calls
  - Supports traditional IMS DL/I database call interface concepts
    - Provides methods to perform DL/I call functions to insert, update, delete, or retrieve data
    - Supports Segment Search Arguments (SSAs)
  - Supports Type 2 and Type 4 connections
  - Provides only LocalTransaction support
    - Single-phase commit

- Benefits
  - Provides Java DL/I API access to IMS database resources for Java applications

157

Use the IMS Universal DL/I driver when you need access to the full set of IMS databases features directly from a Java client in an non-managed environment. You can build segment search arguments (SSAs) and use the methods of the program control block (PCB) object to read, insert, update, delete, or perform batch operations on segments. You can therefore gain full navigation control in the segment hierarchy

Open DataBase – IMS Universal Drivers

- Benefits
  - Provide Java application programs with an open standard API to access IMS databases
    - From a variety of environments
      - Local and Distributed
      - JEE and non-JEE

Open Database Summary

- Online IMS databases can be accessed by -
  - Distributed Java application programs executing anywhere with TCP/IP connectivity
  - ODBA java application programs on any z/OS
  - User-written IMS Connect client application programs executing anywhere with TCP/IP connectivity
  - User-written ODBM client applications in the same IMSplex
- Java programs can use any of the three new Universal Drivers
  - Built on widely used open-standard technologies
- ODBM is a new CSL address space that works with IMS Connect to provide the distributed access facility
- Existing and new ODBA application programs can also use the ODBM address space to manage the ODBA interface, potentially improving the availability of the IMS system

Integration Summary

- IMS Web 2.0 Solution for InfoSphere MashupHub
- IMS TM Resource Adapter 10.2
- IMS 10 Synchronous Callout
- IMS 10 Database Web Services
- IMS 10 Soap Gateway
- IMS 11 Open Database

IMS Version 11

IMS Enterprise Suite

Information Management software

IMS Version 11

IMS Enterprise Suite V1.1

- Part of the IMS SOA Integration Suite
  - Collection of IMS middleware functions and tools
  - Available for download from IMS Enterprise Suite webpage
    - www.ibm.com/software/data/ims/soa-integration-suite/enterprise-suite/
- New packaging method for the most current versions of selected IMS SOA Integration Suite components
  - IMS Enterprise Suite SOAP Gateway
  - IMS Enterprise Suite DLIModel Utility plug-in
  - IMS Enterprise Suite Connect APIs for Java and C
  - IMS Enterprise Suite Java Message Service (JMS) API
- Supports IMS 11 and IMS 10

2

The IMS™ Enterprise Suite, part of the IMS SOA Integration Suite, is a set of components that support open integration technologies to enable new application development and extend access to IMS transactions and data. The IMS Enterprise Suite provides user-friendly standard interfaces, simplifies IMS metadata generation, and enables IMS business event data and monitoring. The IMS Enterprise Suite also eases and expands IMS development (including Java™ and XML), administration, and access. Graphical user interfaces and standards-based programming models are provided through tooling support from the WebSphere® and Rational® product families.

IMS Enterprise Suite V1.1 …

- A no-cost product (5655-T60, S&S 5655-T61)
  - unlimited installs for distributed components
- Provides simplified installation
  - Supports SMP/E for z/OS components
    - Requires z/OS 1.9 for z/OS components
  - Supports IBM Installation Manager for distributed components
- Supports upward compatibility for existing components
  - IMS SOAP Gateway Version 10
  - IMS Version 10 DLIModel Utility plug-in

3

The IMS Enterprise Suite is available for both z/OS® and distributed platforms, and is a no-cost product for unlimited installs. The 5655-T60 is the product number, accompanied by 5655-T61 for Sales and Subscription support (maintenance).

SMP/E provides for easy installation on z/OS as well as Installation Manager support for the components installed on distributed platforms.

Although the components are designed to complement IMS 11, they also support IMS 10. The IMS Enterprise Suite V1.1 is upward compatible from IMS SOAP Gateway Version 10 and the IMS Version 10 DLIModel utility plug-in, allowing existing applications and data to be used without change. Migration and coexistence support is provided for IMS Version 10 users.

IMS Enterprise Suite Connect API

IMS Version 11

IMS Enterprise Suite Connect API

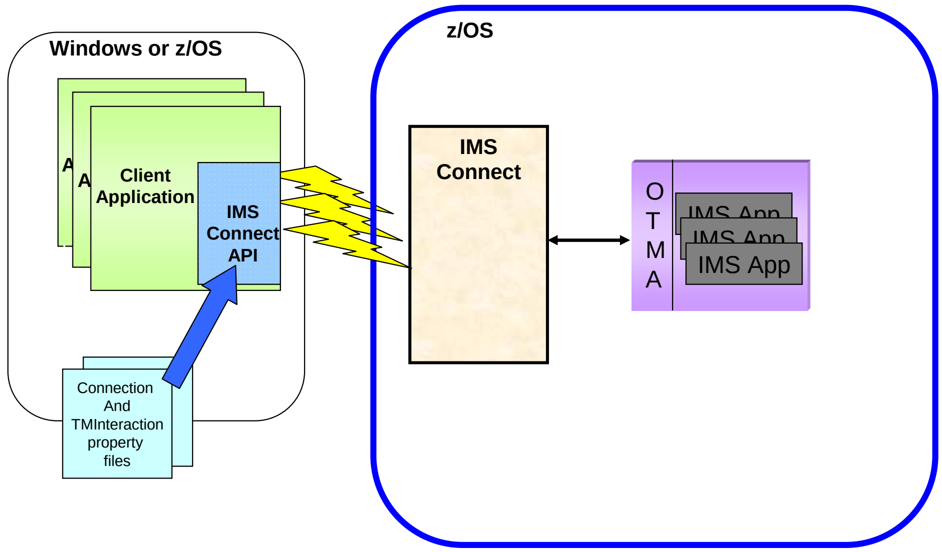
- A simplified callable interface for interaction with IMS Connect
  - Architected on top of the sockets layer
    - Through a set of extensible profiles that define the connections and types of interactions to be performed
      - And a set of both high-level and low-level methods for performing those interactions with IMS Connect

- Addresses the complexities of writing RYO applications
    - Experience in TCP/IP socket programming
    - Understanding the IRM header and the possible flags as well as settings
    - Familiarity with the IMS Connect application protocols and their impact on all the possible types of interactions

5

The new IMS Enterprise Suite Connect API addresses the need for simplification in the way customers interact with IMS Connect. For many releases, remote programmers who have created roll your own (RYO) applications to access IMS resources through IMS Connect have had to understand the complexities of both the IMS Connect IRM header and the associated application protocols for all of the types of interactions. Additionally, these programmers have had to be well versed in TCP/IP socket programming. With the IMS Enterprise Suite Connect API, remote programmers can choose to be shielded from these complexities by taking advantage of a new way of describing connections, interactions, and data to be sent to IMS through a set of re-usable profiles and simple methods. Programmers who want more direct control over certain interactions will continue to have the use of more granular, lower-level calls to interact with IMS Connect and IMS as they do now.

IMS Enterprise Suite Connect API - Overview



6

This visual shows the environment in which the IMS Enterprise Suite Connect API is used. It also depicts the fact that multiple client applications that use the IMS Connect API can be invoked simultaneously. The API communicates with IMS Connect to obtain connection handles as required. In addition to IMS transactions, the initial release of the IMS Enterprise Suite Connect API supports several IMS Connect-supported commands such as the PING and RACF password change commands along with all IMS commands supported by OTMA.

IMS Enterprise Suite Connect API Usage

- Application developer
  - Configures connection and interaction configuration property files read in by API during execution
  - Prepares input data as expected by IMS application and interprets output data as returned by IMS application
    - Single-segment and multi-segment messages
    - Conversational and non-conversational support
  - Manages memory for required data structures used in C implementation
- IMS Enterprise Suite Connect API
  - Generates the IMS Connect input message header
  - Manages interaction according to the IMS Connect message protocols
  - Deals with socket connections made on behalf of the client applications
  - Supports IMS Connect user message exits HWSSMPL0/HWSSMPL1

7

The IMS Connect client application developers are responsible for configuring the connection and interaction properties files read in by API during execution or coding the client applications to set values for these properties during execution. Additionally, the client application has the traditional design responsibilities of preparing the input data, interpreting the output and, for C environments, of managing memory for required data structures.

To shield client applications (and their developers) from the complexities of interacting with TCP/IP sockets and IMS Connect, the IMS Enterprise Suite Connect API both generates the IMS Connect input message header internally and internally manage the interaction with IMS Connect according to the IMS Connect message protocols. In addition, the API deals with the socket connections to IMS Connect made on behalf of the client applications including managing pools of sockets if configured to do so. The API supports all existing IMS Connect user message exits. The default user exit is HWSSMPL0. Customer written user exits are also supported but only if they use the same message and header structures used by HWSSMPL0 and HWSSMPL1.

IMS Enterprise Suite Connect API Usage …

- Optional Features
  - SSL connections
  - API runtime tracing
- Restrictions (not allowed)
  - Two-phase commit (2PC)
  - Resume TPIPE
    - Sync callout support

8

Client applications can also elect to use optional features of the IMS Enterprise Suite Connect API. For example:

- SSL connections for secure, encrypted TCP/IP communications between the API and IMS Connect.

- Different levels of tracing during runtime. With tracing enabled and configured, runtime execution tracing is provided in a manner similar to what is provided by WebSphere Application Server on distributed platforms.

Although the IMS Enterprise Suite Connect API is intended to support all existing IMS Connect functions, existing restrictions include two-phase commit (2PC) interactions as well as Resume TPIPE support and Sync Callout Support.

IMS Enterprise Suite Connect API - Functionality

- Java implementation is available in the initial release
  - C support is being provided through the IMS Enterprise Suite V1.1 service process
- Two design approaches:
  - OO approach for Java (and possible future C++ and/or C# versions)
  - Procedural approach for C
- IMS Enterprise Suite Connect API in Java
  - Supports specification of pathnames to customized files
    - Default connection and interaction properties for the runtime environment
  - OO approach used for Java API is similar to design used for J2EE and JCA
    - Connection factory getConnection call returns Connection interface on which createInteraction is called to return Interaction interface

9

Both Java and C implementations of the IMS Enterprise Suite Connect API will be provided in the initial release. The Java support is already available. Slightly different design approaches are used for each of these languages. An OO approach is supported for object-oriented languages such as Java, C++ and C#, and a procedural approach for procedural languages such as C. The different implementations are supported as similarly as is reasonably possible.

Java client applications may provide pathnames to customized files which specify default connection and interaction properties to be used in that runtime environment.

IMS Enterprise Suite Connect API - Functionality …

- IMS Enterprise Suite Connect API in C
  - API data definitions are supplied as single C-language include file containing
    - Definitions of all IMS Enterprise Suite Connect API constants
    - Definitions of IMS Enterprise Suite Connect API structures and data types
    - Function declarations for external calls
  - Client applications must
    - Specify #include that references the above data definition file
    - Add the location of the IMS Enterprise Suite Connect API C runtime DLL to the path statement for the applicable runtime environment
    - Continue to manage memory, i.e. allocating and freeing memory for C data structures

10

The IMS Connect Enterprise Suite API in C is supplied as an include file, which contains the definition of constants, data types and structures along with function prototypes for each call along with a C DLL which contains the API runtime code. To use the API, the client application must have a #include statement which references the IMS Enterprise Suite Connect API C include file as well as provide accessibility to the provided C runtime DLL by adding its location to the path statement applicable to the runtime environment. Furthermore, the client application continues to be responsible for handling memory management by calling “malloc” and “free” on the data structures. The C implementation is intended to be used in C applications. However, until a separate C++ implementation is available, it can be used in both C and C++ applications.

IMS Enterprise Suite Connect API Calls

- High-level calls
  - Interaction is configured and executed under complete control of the API
    - execute( )
      - Input: input ByteArray*, connection parameters*, interaction parameters*
      - Output: output ByteArray*
        - * Optional parameter
  - "under the covers," issue connect(), send(), Receive() and disconnect() calls as required to execute a complete interaction

11

The IMS Enterprise Suite Connect API allows client applications to interact with IMS Connect at a high level, in which an interaction is configured and executed under complete control of the API, or at a lower level through which the client is able to exert more control over the interaction such as invoking individual calls to open a connection to the target IMS Connect, send a request message and receive a response message over that connection and finally disconnect that connection. Only the high level API is discussed in the class.

The easiest call that applications can use is the high-level execute() call which will, "under the covers," issue connect(), send(), Receive() and disconnect() calls as required to execute a complete interaction.

In Java, the execute() call takes two optional parameters containing the properties of the desired connection and interaction. If unspecified, default values for the connection and interaction properties are used along with an optional one-dimensional (for single-segment input messages) or two-dimensional (for multi-segment input messages) byte array parameter containing the input data. Correspondingly, the call returns a one-dimensional or two-dimensional byte array for either a single-segment or multi-segment reply.

In C, the sendReceive() call takes two optional parameters containing the properties of the desired connection and interaction. Again, if unspecified, default values for the connection and interaction properties are used along with an optional byte array parameter containing the input data and two required output parameters where structures will be returned, one containing the output data and the other containing the call completion code and any error message returned from the API, IMS or IMS Connect. The C sendReceive() call does not have a return value; it returns void.

Configuration

- Configuration of connection and interaction profiles
  - Provides the valid values for users' environment

Connection Profile

Connection socket-related properties
SSL-related properties

Interaction Profile

Interaction-related properties
Message routing-related properties
Security-related properties
Timeout-related properties
Message encoding-related properties
Response-related properties
Tracing-related properties.

The use of the IMS Enterprise Suite Connect API requires the configuration of valid connection and interaction property values appropriate to the runtime environment in which it executes..

Usage Step 1 – First Time Set up

- Basic Flow:
  - Download the IMS Enterprise Suite Connect API archive file from the IMS website and reference according to the language (Java or C) conventions
    - If the client is to run on z/OS, the developer can either use the same distributed approach or use the SMP/E installable files
  - Set up the Connection and Interaction property files
    - The IMS Connect system administrator provides the connection and interaction settings to the IMS Connect client application developer
    - The client application developer records the information into the IMS Connect API Connection and Interaction property files
  - Test the IMS Enterprise Suite Connect API using sample applications that are provided to verify the desired functionality

In this first example, the client application developer sets up the IMS Enterprise Suite Connect API. The flow is as follows:

- The client application developer first downloads the API. The API is bundled as an archive file for download to distributed platforms, and also as SMP/E installable tar file for use on z/OS.

- Then the client application developer obtains the appropriate connection and interaction property values from the systems administrator for the target IMS and IMS Connect systems and saves them into property files for use by the client applications as a way to override the default property values. This can be used to establish installation-specific or environment-specific default property values. The connection property file is used to store information such as hostname, port number, etc. The interaction property file stores information such as commit mode, sync level, reroute name, timeout, codepage, etc.

- Lastly, the IMS Enterprise Suite Connect API administrator or developer can run the sample applications to verify that it is set up correctly

Usage Step 2 – Application Development

- Write an IMS Connect client application using the IMS Enterprise Suite Connect API (high-level calls) to execute an IMS transaction
  - Prerequisites:
    - IMS and IMS Connect are up and running
    - IMS Enterprise Suite Connect API is installed
    - Basic knowledge of IMS Enterprise Suite Connect API
- Basic Flow:
  - Client application prepares the input data
    - Input data can be in single or multiple segments to match the IMS transaction input data structure
      - Client application performs segmentation as necessary
  - The Client application code invokes the high-level execute() method call

14

Once the IMS Enterprise Suite Connect API has been set up, the client application can be coded to use the high-level API and call the execute method to send and receive data to and from IMS.

The flow is as follows:

- The application prepares the input data which can be represent a single or multiple segments depending on the requirements of the IMS Connect user message exit or the target IMS application.

- The application calls the API execute() method

Usage Step 2 – Application Development …

- Basic Flow …
  - As a result of the execute(), IMS Enterprise Suite Connect API performs the following:
    - Obtains a connection handle based on the specified connection parameters
    - Reads the interaction parameters and creates the IRM header
    - Sends the input data to IMS through IMS Connect
    - Receives the output data from IMS Connect
    - Returns the output back to the application
      - Leaves persistent connections open for re-use unless specifically closed
  - The client application verifies the data returned through the IMS Enterprise Suite Connect API
    - Output data can be in single or multiple segments

15

As a result of the execute() call, the IMS Enterprise Suite Connect API performs the following actions:

- Obtains a socket handle based on the specified connection properties.
- Creates the appropriate IRM header using the interaction parameters from the property file
- Sends input data to IMS through IMS Connect.
- Receives output data, if any, from IMS Connect.
- Returns the output back to the application. Note that persistent connections remain open for re-use unless specifically closed.

Finally, the application verifies and processes the data returned, which may contain a single output data segment or multiple output data segments.

Performance and Operational Characteristics

- Operations
  - Sample client application(s) are provided for ease of use
- Performance
  - Goal is to provide a lightweight solution that impacts performance as little as possible
  - No anticipated API performance tuning or planning required
- Security
  - No new or additional security is introduced

16

The sample client application(s) that are provided can be used to validate that the IMS Enterprise Suite Connect API environment has been installed and set up correctly. These samples may also be used as the basis for customer-written client applications.

The goal of the IMS Enterprise Suite Connect API is to provide a lightweight solution that impacts performance as little as possible. No additional actions for API performance tuning or planning are anticipated at this time.

No new or additional security considerations are introduced by the IMS Enterprise Suite Connect API. It simply allows customers to interact with IMS Connect at a higher level.

IMS Enterprise Suite Connect API Benefits

- Benefits
  - Allows client applications to interact with IMS Connect without having to understand the details of the IMS Connect protocol
    - Simplifies design, development and test of the client applications
      - IMS Connect client applications have simple calls (APIs) for quickly and easily interacting with IMS through IMS Connect
        - Allows for easier development of IMS Connect client applications
    - The API manages
      - The TCP/IP communications with IMS
      - The IMS Connect message protocol by sending and receiving the appropriate messages for interactions with IMS Connect
- Runs with IMS 10 or IMS 11

17

The IMS Enterprise Suite Connect API shields client application developers from the complexity of having to understand the detailed functionality of IMS Connect by providing a simple way to request a connection and interaction. The goal of this capability is to make it easier to write new clients to access IMS resources through IMS Connect.

IMS Enterprise Suite SOAP Gateway

IMS Enterprise Suite SOAP Gateway

- IMS Enterprise Suite includes IMS SOAP Gateway with enhancements over the SOAP Gateway provided with previous IMS versions
  - An XML-based connectivity solution that enables existing or new IMS applications to communicate outside of the IMS environment by using SOAP
  - Runs on IMS Version 10
  - New features require IMS Version 11 (with Rational Developer for System z, RDz V7.6)
  - Two limited usage licenses of RDz are provided for each license of IMS11 and IMS 10 (follow the link in the right hand corner of the following page)
    - *http://www-01.ibm.com/software/data/ims/soa-integration-suite/enterprise-suite/*
    - or *http://tinyurl.com/ykyh3uy*
- New features
  - WS-Security
  - Business Event support

19

New SOAP Gateway enhancements use industry standards and enhanced tooling to further extend the reach and participation of your IMS assets in an Service Oriented Architecture (SOA).

Two limited usage licenses (without support) of Rational Developer for System z are provided for each license of IMS Version 11 and IMS Version 10. Terms and conditions are outlined in the Rational Developer for System z licensing along with Rational Developer for System z on the IMS Enterprise Suite SOAP Gateway download Web site.

IMS Enterprise Suite SOAP Gateway: WS-Security

- Capabilities
  - Supports dynamic Web Services Security as part of the SOAP envelope
    - Dynamic authentication on a per-message basis for accessing IMS applications as Web services
      - Based on the IMS Connect security setup
    - Previous authentication was static
      - Same userid for all accesses on a web-service connection between the IMS SOAP Gateway and IMS Connect
- Benefits
  - SOAP Clients can send WS-Security header for authentication to IMS SOAP Gateway for access to IMS applications

20

IMS Enterprise Suite SOAP Gateway provides support for dynamic Web Services Security thereby supporting dynamic security information on a per message basis instead of on a per connection basis (the static support that IMS provided in earlier versions of the IMS SOAP Gateway). This support is provided through the headers and body as part of the SOAP envelope. To take advantage of this authentication capability, client applications will need to send in the WS-Security header information.

Business Event Support

- Capabilities
  - Enables business event processors (e.g., WebSphere Business Events V6.2) and monitors (e.g., WebSphere Business Monitor V6.2) to receive business event data from IMS applications
    - Application uses ISRT to ALTPCB to OTMA hold queue
    - OTMA destination descriptor routes message through SOAP Gateway
    - RDz is used to generate the correlator file, XML converter and data mapping XSD file
    - SOAP Gateway callout thread pulls message with event data from OTMA hold queue
- Benefits
  - Enables IMS applications participate with products that support business activities monitoring
    - Uses standard interfaces with HTTP/SOAP transport

21

IMS SOAP includes Business Event support, which enables WebSphere® Business Events V6.2 and WebSphere Business Monitor V6.2 to receive business event data from IMS applications for business activities monitoring. This helps IMS users leverage existing IMS assets to explore business event processing and to explore business event monitor solutions, identify business problems, correct exceptions, and change processes to help increase business competitiveness by improving process efficiencies. Using the graphical tooling interface and runtime engines in WebSphere Business Events and WebSphere Business Monitor, business users are empowered to define and manage business events, proactively monitor IMS business activities, instrument IMS business logic, identify new business opportunities, and mitigate risks.

The steps to enable an IMS application to emit a business event to be processed or monitored by a business event processing engine are similar to the steps in the one-way asynchronous callout scenario, with a few minor differences that are specific to WebSphere® Business Events and WebSphere Business Monitor.
1.Use the ISRT ALTPCB call to place the event data on the OTMA hold queue.
2.Use an OTMA destination descriptor for an IMS application to route business event data to a server that is accessible to SOAP Gateway without the need to code assembler routing exits.
3.Generate the correlator file, the XML converter, and the data mapping XSD file by using Rational® Developer for System z® from the IMS application source file.
4.Deploy the XML converter in IMS Connect.
5.Create a connection bundle by using the SOAP Gateway deployment utility with the required information for SOAP Gateway to connect to IMS Connect.
6.Configure the IMS Enterprise Suite SOAP Gateway server to route the business events to the specified business event processing server. Configuration is done by using the SOAP Gateway deployment utility and by providing the WSDL or XSD file for the business event server.
7.Start the callout thread. The SOAP Gateway callout thread is used to pull the business event data from the OTMA hold queue.

IMS Enterprise Suite DLIModel Utility Plug-in

IMS Enterprise Suite DLI Model Utility Plug-in

- An Eclipse plug-in that can be installed with IBM Eclipse SDK or shell-shared with existing Rational Application Developer Software or Rational Developer for System z
  - Transforms your IMS DB information (PSBs, DBDs, COBOL copybooks, PL/I includes)
    - Into application-independent metadata that can be used for IMS XML or Java application development in an Eclipse-based environment
- Other versions
  - IMS DLIModel utility plug-in
    - Web downloadable version that runs as a plug-in to Eclipse, Rational Application Developer (RAD), and Rational Developer for System z (RDz)
  - DLIModel utility
    - IMS-shipped version that runs from Unix System Services or from the z/OS BPXBATCH utility
    - IMS 10 is the last level of IMS to include this IMS-shipped version
- IMS 11 does not include the DLIModel Utility

23

IMS Version 10 is the last release of IMS to include in its base the IMS-shipped version of the DLIModel utility. To take advantage of the most current DLIModel utility plug-in features, customers using the IMS Version 10 DLIModel utility should migrate to the IMS Enterprise Suite DLIModel utility plug-in, which is part of the IMS Enterprise Suite V1.1.

IMS Enterprise Suite DLI Model Utility Plug-in …

- Enhancements include:
  - Support for the import of PL/I includes
  - Addition of PROCOPT to the DLIDatabaseView metadata class
    - Enhances the JDBC driver performance
  - Support for a graphical view of Virtual Foreign Key fields
    - Allows JDBC programmers to understand the relational view of the data
      - Used with Universal JDBC driver and Universal DB resource adapter
  - Enhanced wizard to automatically selects DBDs referred to by PSBs
  - Addition of search, save and print functions to the graphical editor for PSBs and DBDs
  - Seamless shell-sharing with other Eclipse-based IBM products

24

The IMS Enterprise Suite DLIModel utility has enhancements over the DLIModel utility provided with IMS 10 and previous versions of IMS. The plug-in provides a simplified interface for transforming IMS database information into metadata for use with application development. Enhancements help ease the data transformation, data integration, and tooling installation for IMS DBA, system or application programmer by:

- Allowing PL/I 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 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 to automatically select database descriptors (DBDs) referred to by a program specification block (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.

JMS API for Synchronous Callout

JMS API for Synchronous Callout

- Java Message Service (JMS) API support for synchronous callout
  - May be used with IMS 10 or IMS 11
  - Used with Java Dependent Regions (JMPs and JBPs)
  - Uses ICAL for synchronous callout
  - JMS API is a JAR file in the classpath

26

JMS API expands Java application development in Java dependent regions to offer synchronous callout support via the ICAL DL/I call.

The JMS API is a JAR file that must be present in your classpath to support the use of the Java API for callout in IMS Version 10 and Version 11. The Java API for callout enables IMS applications to issue synchronous callout requests to external services from within a Java message processing (JMP) or Java batch processing (JBP) application.

IMS Enterprise Suite - Summary

- The IMS Enterprise Suite consists of components that are designed to support open integration technologies to enable new application development and extend the access to IMS transactions and data
  - The IMS Enterprise Suite is available for both z/OS and distributed platforms
  - Installation instructions can be found at:
    - *http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp?topic=/com.ibm.ims.es.doc/ies_installcontainer.htm*

      or
    - *http://tinyurl.com/ygovofz*

IMS Version 11

DBRC

Information Management software

DBRC

- BPE Based DBRC Region
- Unconditional Deletion of RECON Information
- Security Override for Non-production RECON Copies
- Change Accumulation Enhancement (SPE for IMS 10)
- DELETE.DB Performance Enhancement (SPE for IMS 10)
- DBRC Migration

BPE Based DBRC Region

BPE Based DBRC Region

- BPE for online system DBRC region
  - Optional
  - BPE provides improved tracing
  - BPE provides improved user exit management
  - Only applies to online DBRC region
    - Not used for batch jobs, IMS utilities, or DBRC utility

4

IMS 11 introduces the use of a BPE address space for the online DBRC region. This use is optional, however, it adds capabilities that do not exist without it. The use of BPE improves the tracing options and user exit management for DBRC. BPE is only used by DBRC for online regions. It is not used by IMS batch jobs, IMS utilities, or the DBRC utility.

IMS Version 11

BPE Based DBRC Region

- Sample JCL for use of BPE

```
//DBRC     PROC RGN=0M,SOUT=A,RESLIB='IMS.SDFSRESL',BPECFG=BPECONFG,DBRCINIT=000,PARM1=
//DBRCPROC    EXEC PGM=BPEINI00,REGION=&RGN,
//    PARM='BPECFG=&BPECFG,BPEINIT=DSPBINI0,DBRCINIT=&DBRCINIT,&PARM1'
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//         DD  DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD  DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT
//JCLOUT   DD SYSOUT=(A,INTRDR)
//JCLPDS   DD DSN=IMS.SKELJCL.PROCLIB,DISP=SHR
//IMSDALIB DD DSN=IMS.DYNALLOC,DISP=SHR
//SYSABEND DD SYSOUT=&SOUT
```

  - BPECFG= specifies BPE configuration member
  - DBRCINIT= specifies DBRC initialization member (DSPBIxxx)
- Sample PROC DSPBPROC is provided

5

As in previous releases, the DBRCNM on the IMS execution parameter specification is used to determine the DBRC procedure that will be started. If DBRC is not already started, IMS will issue an MVS start command for the procedure specified by DBRCNM. This is the same process that IMS uses with or without a BPE based DBRC region.

The sample JCL shown here may be used as the DBRC procedure for the execution of DBRC using BPE. The user specifies two PROCLIB members. THE BPECFG= parameter specifies the BPE parameters. The DBRCINIT= parameter species the DBRC initialization member which contains some DBRC parameters. These are shown on the next page. The BPEINIT= parameter on the EXEC statement always specifies DSPBINI0. This is the DBRC supplied initialization module. The DBRC initialization member is DSPBIxxx where 'xxx' is the value specified in the DBRCINIT parameter.

A sample DBRC BPE startup PROC is provided. It is DSPBPROC.

BPE Based DBRC Region

- DBRC Initialization PROCLIB member (DSPBIxxx)
  - Parameters
    - IMSPLEX=, DBRCGRP=
      - IMSplex and DBRC group parameters used with SCI
      - The use of an exit routine to specify these is recommended.
    - VSAMBUFF=(indexbuffers,databuffers)
      - VSAM LSR pool specification
      - Not used with PRA
      - Replaces DSPBUFFS for the online region
        - DSPBUFFS is used for other DBR regions

6

The DBRCINIT parameter in the procedure is used to specify the last three characters of the DBRC initialization parameters PROCLIB member (DSPBIxxx). The parameters in this member are shown here.

The IMSPLEX and DBRCGRP parameters correspond to the same parameters on the DBRC procedure used without BPE. IMSPLEX specifies the five-character IMSplex group name.

DBRCGRP specifies the 1- to 3-character identifier (ID) assigned to a group of DBRC instances that access the same RECON data set in an IMSplex. Typically, these parameters are set by the DBRC SCI Registration exit routine (DSPSCIX0).

VSAMBUFF= provides as alternative way of specifying the number of index and data buffers in the VSAM LSR pool for this instance of DBRC. Without this parameter, DSPBUFFS is used. Using the VSAMBUFF= parameter eliminates the need to assemble and bind the DSPBUFFS CSECT in order to specify the number of buffers.

BPE Based DBRC Region

- DBRC Initialization PROCLIB member example:

```
/*****************************************************************/
/* DBRC INITIALIZATION PROCLIB MEMBER                            */
/*****************************************************************/
VSAMBUFF=(80,160)      /* 80 index buffers, 160 data buffers */
```

7

This shows an example of the DBRC Initialization Parameters PROCLIB member.

The VSAMBUFF parameter is used to specify the LSR buffers used for the RECONs. The IMSPLEX and DBRCGRP parameters are not included in this example. The use of the DBRC SCI Registration exit is recommended for establishing these values.

BPE Based DBRC Region

- DBRC tracing with BPE
  - BPE configuration PROCLIB member parameters
    - TRCLEV=(ERR,HIGH,DBRC,PAGES=num_pages)
      - Trace of errors in DBRC address space
    - TRCLEV=(RQST,level,DBRC,PAGES=num_pages)
      - DBRC request processing trace
    - TRCLEV=(MODF,level,DBRC,PAGES=num_pages)
      - Module flow trace
    - TRCLEV=(GRPS,level,DBRC,PAGES=num_pages)
      - DBRC group services message and notification trace

8

The BPECFG parameter in the procedure is used to specify the last three characters of the BPE configuration PROCLIB member. The TRCLEV parameters in this member control tracing. This is true for all users of BPE. The parameters are the standard ones used for BPE tracing. The first parameter identifies the type of trace. There are four traces for DBRC. They trace errors in the DBRC address space, DBRC request processing, DBRC module flow, and DBRC group services messages and notifications. The second parameter controls the level of the trace. It may be LOW, MEDIUM, or HIGH. The third parameter identifies the component being traced. DBRC is identified by "DBRC" in this parameter. The PAGES= parameter specifies the number of pages for the trace. This default is 2 pages for the ERR trace and 8 for the other DBRC traces.

The error trace level is always HIGH. The value specified for the error trace is ignored.

BPE Based DBRC Region

- User exit management
  - BPE manages RECON I/O, DBRC Security, and Statistics exit routines
    - "DBRC Security" is new name for "DBRC Command Authorization" exit
  - Exits may be updated without bringing down DBRC and IMS online system
  - More than one exit routine of each type may be used
  - DBRC statistics are available to the BPE statistics exit
    - Uses standard BPE Statistics user-supplied exit interface
      - DSPBSTX: Header containing number of clients
      - DSPBST1: Performance information including VSAM I/Os and RESERVEs
      - DSPBST2: Statistics about request types for each DBRC function
  - All exits use BPE exit interface
    - RECON I/O and DBRC Security must use this interface
      - Previous interface must be used for non-BPE invocations (e.g. batch and utilities)

9

One of the advantages of using BPE is that DBRC benefits from BPE's user exit management. This applies to the RECON I/O exit (previously DSPDCAX0), the DBRC Security exit (previously called the DBRC Command Authorization exit, DSPCEXT0), and the new statistics exit. Exits managed by BPE are refreshable. New version of them may be implemented without terminating the DBRC address space and the control region. Multiple exit routines may be specified for the same exit. This might be needed if multiple IMS tools (products) each required an exit routine.

When using BPE the exit routines may have any name. They are not limited to the names previously dictated for the RECON I/O exit routine (DSPCEXT0) and the DBRC Security exit routine (DSPDCAX0).

There is a new statistics exit capability. BPE has a standard BPE Statistic user-supplied exit interface. This interface is used. Exit routines are passed a header containing the number of clients and a pointer to the DSPBST1 control block. This control block contains performance information. This includes the number of VSAM I/Os by type (GET, PUT, and ERAE), the number of RESERVEs, and the time spent waiting on RESERVE requests. DSPBST2 contains statistics that are related specific DBRC request types such as ALLOCATE, AUTHORIZE, and SIGNON. The statistics in both DSPBST1 and DSPBST2 are reset to zero after the exit is called. This means that the data in accumulated between calls to the exit, but not kept across calls to the exit.

The exit interface is different when using BPE. This means that any RECON I/O exit routine and or DBRC Security exit routine previously used must be modified to be invoked under BPE. Since these exits are probably required for DBRC executions not using BPE, such as batch jobs and utilities, the non-BPE interface must be maintained also. There are two techniques that may be used to accommodate this. First, the BPE exits may have a small front end that calls the exit routines using the non-BPE interface. Second, the exit routines may be written to detect the interface being used. New sample RECON I/O and DBRC Security exit routines, DSPCEXT1 and DSPDCAX0, are shipped with IMS 11. These new samples show how to recognize the BPE and non-BPE parameter lists.

BPE Based DBRC Region

- BPE configuration PROCLIB member example

```
/**************************************************************/
/* BPE CONFIGURATION PROCLIB MEMBER FOR DBRC                  */
/**************************************************************/
LANG=ENU             /* LANGUAGE IS ENGLISH                   */
/**************************************************************/
STATINV=600          /* STATS USER EXIT INTERVAL IS 600 SEC. */
/**************************************************************/
/* 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 dbrc traces                                */
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 */
/**************************************************************/
/* User exit list PROCLIB member specifications               */
EXITMBR=(BPEEXIT0,BPE)         /* BPE USER EXIT DEFINITIONS  */
EXITMBR=(DSPEXIT0,DBRC)        /* DBRC USER EXIT DEFINITIONS */
/**************************************************************/
```

10

This is an example of a BPE configuration PROCLIB member for use with a DBRC region. The PROCLIB member is specified in the BPECFG= execution parameter.

LANG=ENU must be specified. English is the only supported language.

STATINV= is an optional statement. It controls the interval between the driving of the statistics exit routines.

In this example, the first three TRCLEV= statements are for BPE system traces. The last four TRCLEV= statements specify DBRC traces. The default pages specification is overridden for the DBRC request trace.

The two EXITMBR= statements specify PROCLIB members for the BPE and DBRC exit routine definitions.

BPE Based DBRC Region

- BPE user exit list PROCLIB member example

```
/*****************************************************************/
/* BPE USER EXIT LIST PROCLIB MEMBER                             */
/*****************************************************************/
/* Define one BPE init/term exit routine: MYINIT00.
*/
/*****************************************************************/
EXITDEF(TYPE=INITTERM,EXITS=(MYINIT00))
/*****************************************************************/
/* Define 2 BPE Statistics exit routines: HHGSTAT0 and MYDSTAT0  */
/* MYDSTAT0 processes DBRC statistics                            */
/*****************************************************************/
EXITDEF(TYPE=STATS,EXITS=(HHGSTAT0,MYDSTAT0))
```

11

This is an example of a PROCLIB member which specifies the BPE user exits. The name of this PROCLIB member is specified in the EXITMBR=(member name,BPE) statement in the BPE configuration PROCLIB member.

In this example, there are two BPE exits. The first is the initialization/termination exit. Routine MYINIT00 is specified for this exit. The second exit is the BPE statistics exit. It has two exit routines specified. They are HHGSTAT0 and MYDSTAT0. These exit routines have access to the DBRC statistics in the DSPBSTX, DSPBST1, and DSPBST2 control blocks. The comments in the member indicate that MYDSTAT0 will process this information.

BPE Based DBRC Region

- DBRC user exit list PROCLIB member example

```
/******************************************************************/
/* DBRC USER EXIT LIST PROCLIB MEMBER                             */
/******************************************************************/
/* DEFINE 1 RECON I/O EXIT: ZDBRCIO0                              */
/* WITH AN ABEND LIMIT OF 8.                                      */
/******************************************************************/
EXITDEF(TYPE=RECONIO,EXITS=(ZDBRCIO0),ABLIM=8,COMP=DBRC)
/******************************************************************/
/* DEFINE 1 DBRC SECURITY EXIT: ZDBRCSE0                          */
/******************************************************************/
EXITDEF(TYPE=SECURITY,EXITS=(ZDBRCSE0),COMP=DBRC)
```

12

This is an example of a PROCLIB member specifies the DBRC user exits. The name of this PROCLIB member is specified in the EXITMBR=(member name,DBRC) statement in the BPE configuration PROCLIB member.

In this example, the RECON I/O exit routine name is ZDBRCO0. When DBRC is not run in a BPE address space, the RECON I/O exit routine must be named DSPCEXT0. When a BPE address space is used, there may be multiple RECON I/O exit routines and they may have any valid name.

In this example, the DBRC Security exit routine name is ZDBRCSE0. The DBRC Security exit routine was called the DBRC Command Authorization exit routine in previous IMS releases. When DBRC is not run in a BPE address space, the DBRC Security exit routine must be named DSPDCAX0. When a BPE address space is used, there may be multiple DBRC Security exit routines and they may have any valid name.

BPE Based DBRC Region

- New messages:

```
DSP2001I DBRC READY
DSP2003A INVALID VALUE SPECIFIED FOR keyword PARAMETER
DSP2005I DBRC SHUTDOWN COMPLETE
```

13

The BPE based DBRC region issues new and changed messages. You will see the DSP2001I and DS2005I messages at startup and shut down. You may also see the DSP2003A message. There is also a DSP2002E message for initialization errors.

The DSP1154A message may have different text since it lists the name of the DBRC Command Authorization exit routine. Previously, this was always DSPDCAX0. Now, it may have a user specified name.

BPE Based DBRC Region

- Benefits
  - Improved user exit management
  - New statistics exit
  - Improved tracing capabilities
  - Simplified definition of LSR buffers for the RECONs

14

The BPE based DBRC region for online systems delivers several benefits.

It improves the management of user exits. It provides a new statistics exit. That is, one can use the BPE statistics exit to gather DBRC statistics.

racing capabilities are improved when using a BPE based region.

It simplifies the definition of LSR buffers. Of course, this only applies when Parallel RECON Access is not used because these buffers are not used with PRA.

Other DBRC Enhancements

Unconditional Deletion of RECON Information

- CLEANUP.RECON command
  - Used to delete old information in RECONs
    - PRILOG, IC, ALLOC, REORG, and RECOV information
    - If time not specified, the log retention time (LOGRET) is used
  - Information deleted even though IC requirements are not met
    - GENMAX and RECOVPD are ignored
  - Logs are deleted if closed before the specified time
    - Other logs are compressed in the PRILOG records

```
CLEANUP.RECON RETPRD(time_interval) | TIME(time_stamp)
              DBONLY LASTIC DBRANGE(first_db,last_db)
              LISTDL | NOLISTDL
```

16

This enhancement addresses the situation where information about logs and image copies exist for data sets that no longer physically exist, most likely due to expiration policies. It also helps to identify databases that are potentially no longer being used. Some users want to delete records from the RECONs even though the rules used by DBRC would not delete them. They want to do this primarily because they may have old information about database data sets and they do not have image copy records that old. For example, a user may fail to take an image copy of a database data set for many weeks. They may have an alternative way of recovering or restoring the database. In the meantime, the image copy data set for the oldest image copy recorded in the RECONs may have been deleted. DBRC's rules are designed to keep recovery information for a database data set that may be used with the oldest Image Copy for the database data set. This prevents the no longer usable information from being deleted from the RECONs. This enhancement makes it easy for users to delete old recovery information from the RECONs in these situations.

DBRC's rules provide that when the RECON data set is notified of an image copy, it may delete or reuse the oldest in-use IC record, and a later IC record becomes the oldest IC record. RECOV and REORG records with start times earlier than the (now) oldest IC record, and ALLOC records with DEALLOC times earlier than that, are now extraneous, and are deleted from the RECON data set. This is the image copy cleanup process. At the same time that extraneous IC records are deleted from the RECON data set, all active ALLOC records are updated to the time of the first log volume necessary for recovery, based on the oldest image copy for the DBDS or area. When the cleanup process deletes an extraneous ALLOC record it changes the state of the associated LOGALL records. Once all the ALLOC records associated with the LOGALL record have been deleted (this may take place over many image copies for many databases), the PRILOG record associated with the LOGALL record becomes inactive. When a PRILOG record is inactive, the PRILOG may be compressed. PRILOG record compression is the deletion of all inactive data set entries in the PRILOG record. A data set entry is defined as being inactive when it is older than all of the following criteria: (1) log retention period, (2) oldest allocation (ALLOC) for any database updated on that log and (3) earliest restart checkpoint for the online IMS. The CLEANUP.RECON command deletes PRILOG records which have no data set entries with times later than the specified deletion time.

Unconditional Deletion of RECON Information

- CLEANUP.RECON command

```
CLEANUP.RECON RETPRD(time_interval) | TIME(time_stamp)
              DBRANGE(first_db,last_db) DBONLY LASTIC
              LISTDL | NOLISTDL
```

  - All parameters are optional
  - RETPRD: Data older than RETPRD is deleted
  - TIME: Data with timestamps before TIME is deleted
  - If RETPRD and TIME are not specified, data older than LOGRET for RECONs is deleted
  - DBONLY: Limits actions to IC, ALLOC, REORG, and RECOV records
    - PRILOG information is not deleted
  - LASTIC: Allows the deletion of the last IC record
    - If not specified, the last IC record for a database data set is not deleted
    - IC Needed flag is set when last IC record is deleted
  - DBRANGE: Limits actions to specified set of databases
    - Used to limit the work done by one issue of the command
  - LISTDL or NOLISTDL: May be used to override the default in the RECON header
    - Determines if data set names deleted from the RECON are listed in the output

17

You may specify either RETPRD, TIME, or no time value. RETPRD specifies a length of time or "aging value". For example, a specification of 2 months would mean that records more than two months old should be deleted. TIME specifies a timestamp. For example, a specification of January 15, 2010 would mean that records for times before January 15, 2010 should be deleted. If neither RETPRD nor TIME is specified, the log retention time is used. Log retention time is set by the LOGRET parameter in the INIT.RECON or CHANGE.RECON command. It defaults to 1 day (24 hours). Times are specified in DBRC's timestamp formats. For example, '10 12:30' means 10 days, 12 hours, and 30 minutes and '15' means 15 days.

DBONLY limits the actions of the command. When specified, only database related records (IC, ALLOC, REORG, and RECOV) are deleted. PRILOG records are not compressed or deleted.

LASTIC allows the deletion of the last IC record for a database data set. If the last IC for a data set is deleted, the IC Needed flag is turned on. If LASTIC is not specified, the command will not delete all IC records for a database data set.

DBRANGE is used to specify a range of database names for which the cleanup is done. These are names of HALDB partitions and non-HALDB databases including DEDBs. HALDB master databases within the range are not processed. HALDB partitions within the range are processed. The names specified do not have to correspond to databases or partitions registered in the RECONs. For example, if first_db is LAAAAAAA and it is not registered, the processing will start with the first database after LAAAAAAA. If DBRANGE is included, but first_db is not specified, processing begins with the first database. If DBRANGE is included, but last_db is not specified, processing continues through the last database.

LISTDL and NOLISTDL may be specified to override the default for listing deleted data sets. The INIT.RECON and CHANGE.RECON commands allow you to set the default to LISTDL or NOLISTDL. This may be overridden on the CLEANUP.RECON or DELETE.LOG command. LISTDL specifies that the data set names deleted from the RECON are listed in the output.

Like other DBRC commands, the CLEANUP.RECON command is protected by DBRC Command Authorization.

Unconditional Deletion of RECON Information

- Messages from CLEANUP.RECON command
  - Messages indicate that log data set information was deleted

```
DSP1047I DELETED DSN=data_set_name, FILESEQ=nnnn,VOLSER=vvvvvv(,...)
```

  - Messages indicate that recovery related information was deleted for database or area

```
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=dbname DDN=ddname

DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=dbname AREA=areaname
```

    - If LISTDL is in effect, additional information will include identification of image copy, ALLOC, RECOV, and REORG records

Unconditional Deletion of RECON Information

- Recommendation
  - Reorganize the RECONs after using the CLEANUP.RECON command
    - Reorganization will reclaim space
- Benefits
  - Simplifies maintenance of RECONs
  - Useful when data sets are deleted by expiration policy
    - Data sets older than a specified age are automatically deleted
  - May be used to identify data sets which are no longer in use
    - Command may be issued for a copy of the production RECONs

19

After issuing the CLEANUP.RECON command, it may be advisable to reorganize the RECONs. This will reclaim the space create by the deletion of many records.

The CLEANUP.RECON command simplifies the maintenance of RECONs for some users. When a data set expiration policy deletes data sets older than a specified age, there may still be information about these data sets in the RECONs. This information is not useful, since these data sets are no longer available for database recovery. A CLEANUP.RECON command with a RETPRD equal to the expiration period will delete information about the deleted data sets from the RECONs.

The command may also be used to identify data sets which are no longer in use and for which image copies are no longer taken. In circumstances such as these, it may be wise to copy the RECONs and issue the command for the copy. This will identify the data sets without deleting their information from the production RECONs.

Security Override for Non-production RECON Copies

- New option on INIT.RECON and CHANGE.RECON commands
  - Allows processing of RECON copies without invoking DBRC command authorization security
  - Security enforced only when COPY1 data set name includes specified substring

```
INIT.RECON ... CMDAUTH(SAF|EXIT|BOTH|NONE,safhlq,rcnqual)

CHANGE.RECON ... CMDAUTH(SAF|EXIT|BOTH|NONE,safhlq,rcnqual)
```

- Security enforced if the *rcnqual* value is a substring in the data set name of COPY1
  - If rcnqual is specified with an '*' at the end and in quotes, data set name must begin with rcnqual value
  - Warning message issued if CMDAUTH is in effect but *rcnqual* does not match

```
DSP1211W THE CURRENT COMMAND AUTHORIZATION SETTING WILL NOT BE ENFORCED
```

20

Some users have asked for a way to override the use of DBRC command authorization security on copies of the RECON data sets. This would be particularly useful for problem investigations where production RECONs are copied and the analysis is done against a copy. When DBRC command authorization security is used, each command must pass SAF or an exit routine's security check. Of course, one could turn off security by changing the first CMDAUTH parameter to NONE, but this would require authorization to issue the CHANGE.RECON command. IMS 11 provides a simpler solution.

IMS 11 allows users to specify a RECON qualifier (rcnqual) for the data set name of the current COPY1 data set. DBRC command authorization security is only enforced if the data set name for COPY1 includes this string of characters. The string could be the data set name, a high-level qualifier of the data set name, or any other part of the data set name. When copies of the RECONs are made, if the data set names of the copies do not include the RECON qualifier, security is not checked. This allows problem analysts to use commands against the copies without authorizing them for commands against the production RECONs.

When security is defined for the RECONs (SAF, EXIT, or BOTH is specified) but the RECON qualifier (rcnqual) is not part of the COPY1 data set name, message DSP1211W is issued. This warns the user that command authorization security is not being enforced.

The INIT.RECON and CHANGE.RECON commands cannot set the 'rcnqual' value to something other than a substring of the current COPY1 data set name. This means that the 'rcnqual' setting cannot be used to override security on the RECONs directly. It controls the use of security when the RECONs are copied or renamed. The following message is issued if the 'rcnqual' value is not a substring of the current COPY1 data set name.

DSP1210E THE RECONHLQ SPECIFIED IS NOT A SUBSTRING OF THE RECON DSN

Security Override for Non-production RECON Copies

- First example of use:

```
CHANGE.RECON ... CMDAUTH(SAF,PRDRCN,IMSG.PROD)
```

  - Production RECONs
    - Production RECON data set names:

```
RECON1: IMSG.PROD.RECON1
RECON2: IMSG.PROD.RECON2
RECON3: IMSG.PROD.RECON3
```

    - Security is enforced since "IMSG.PROD" is in data set name
  - Copies of RECONs are made for problem investigation
    - Data set names for RECON copies:

```
RECON1: IMSG.COPY.RECON1
RECON2: IMSG.COPY.RECON2
RECON3: IMSG.COPY.RECON3
```

    - Security is not enforced since "IMSG.PROD" is not in data set name
      - Problem analyst can issue commands against the copies

21

This is an example of how the security override could be used. The CHANGE.RECON command is used to set the RECON qualifier to 'IMSG.PROD'. This means that DBRC command authorization security is only enforced when this substring is part of the data set name for the COPY1 RECON. When a problem occurs with the production RECONs, they are copied. The copies have data set names with 'IMSG.COPY', not 'IMSG.PROD'. Since 'IMSG.PROD' is not included in the data set name, DBRC command authorization security is not enforced. The problem analyst does not have to be granted authority to issue DBRC commands.

The CHANGE.RECON command to set the RECON qualifier does not have to be issued before the problem occurs. There are two ways to set the RECON qualifier. First, one may issue the CHANGE.RECON command for the production RECONs before he copies are made. Second, someone with authority to issue the CHANGE.RECON command may issue the command on the copies.

Security Override for Non-production RECON Copies

- Second example of use:

```
CHANGE.RECON ... CMDAUTH(SAF,PRDRCN,'IMSG.PROD*')
```

  - Production RECONs
    - Production RECON data set names:

```
RECON1: IMSG.PROD.RECON1
RECON2: IMSG.PROD.RECON2
RECON3: IMSG.PROD.RECON3
```

    - Security is enforced, "IMSG.PROD" is high level qualifier for data set name
  - Copies of RECONs are made for problem investigation
    - Data set names for RECON copies:

```
RECON1: SMITH.COPY.IMSG.PROD.RECON1
RECON2: SMITH.COPY.IMSG.PROD.RECON2
RECON3: SMITH.COPY.IMSG.PROD.RECON3
```

    - Security is not enforced, "IMSG.PROD" is not high level qualifier of name
      - Problem analyst can issue commands against the copies

22

This is another example of how the security override could be used. The CHANGE.RECON command is used to set the RECON qualifier to 'IMSG.PROD*'. This means that DBRC command authorization security is only enforced when this substring is the high order part of the data set name for the COPY1 RECON. When a problem occurs with the production RECONs, they are copied. The copies have data set names with 'SMITH.COPY' as a prefix. Since 'IMSG.PROD' is not the high order part of the data set name, DBRC command authorization security is not enforced. The problem analyst does not have to be granted authority to issue DBRC commands.

Security Override for Non-production RECON Copies

- Recommendation
  - Specify the RECON qualifier when migrating to IMS 11
- Benefits
  - Simplifies the handling of security for copies of RECONs
    - Especially important when shipping copies outside your installation
      - Others do not have to be given security information to investigate a problem
    - Problems could be resolved faster

When you migrate to IMS 11 is a good time to specify the RECON qualifier. This avoids the need to change it later when you are doing problem determination.

The primary benefit of this enhancement is the simplification of security handling when the RECONs need to be examined to address a problem. Typically, the people investigating the problem do not have the authority to issue DBRC commands for this set of RECONs. When a copy is made, the same security rules apply to it unless this enhancement is used. By limiting enforcement to those RECONs with the production data set names, problem investigators may more easily and quickly examine the RECONs.

Change Accumulation Enhancement (IMS 10 SPE)

IMS 10 APAR: PK53223

- Allow changes in RECON contents between the time the CA JCL is generated and executed
  - Changes allowed by IMS 10 SPE:
    - New Image Copy
    - OLDS archived which changes the purge time
    - HALDB Online Reorganization
    - Offline Reorganization
  - Previously, theses changes would cause a utility execution failure
  - DBRC will allow the JCL generated for Change Accumulation to execute using the purge time in the JCL as long as input logs verify

- Benefit
  - Image copies, archives, and reorgs will not cause Change Accum failures

24

A Small Programming Enhancement (SPE) has enhanced the usability of the Change Accumulation utility for IMS Version 10. The SPE allows some changes in the content of the RECONs between the time that the GENJCL.CA produced the Change Accumulation utility JCL and the time that it is run. Previously, a change in the purge time for the database data set or a reorganization would cause the utility to fail. The purge time could be changed by taking an Image Copy after the GENJCL execution or by the archiving of an OLDS which causes a concurrent (fuzzy) image copy to be usable.

With this SPE applied to an IMS 10 system, the Change Accumulation utility will execute using the purge time in the JCL as long as the input logs verify at execution time.

This provides greater flexibility and usability in the use of Change Accumulation.

These changes have cause changes in the messages issued by the Change Accumulation utility.

Change Accumulation Enhancement (IMS 10 SPE)

- New informational messages from Change Accumulation utility
  - DSP0714I SPECIFIED PURGETIME IS EARLIER THAN THE CURRENT PURGETIME FOR DBDNAME=dbdname DDNAME=ddname CURRENT PURGETIME timestamp IS NOT USED
  - DSP0716I LATER OFFLINE REORGANIZATION FOLLOWING PURGETIME FOR DBDNAME=dbdname DDNAME=ddname RUNTIME=timestamp
    - RC00 from Change Accumulation Utility if one of these new informational messages is issued

25

Two new informational messages may be issued by the Change Accumulation utility. These are likely to be issued when the JCL is created by GENJCL.CA and either an Image Copy or an offline reorganization is done before the JCL is used.

When these informational messages are issued a return code of 00 is returned by the utility.

Change Accumulation Enhancement (IMS 10 SPE)

- New error messages from Change Accumulation utility for failures
  - `DSP0715E SPECIFIED PURGETIME IS LATER THAN THE CURRENT PURGETIME FOR DBDNAME=dbdname DDNAME=ddname CURRENT PURGETIME = timestamp`
    - Previously, DSP0711I was issued when the purge time was not the current image copy time. DSP0711I is no longer issued.
  - `DSP0717E CHANGES INCLUDED AFTER OFFLINE REORGANIZATION FOR DBDNAME=dbdname DDNAME=ddname`
    - Replaces `DSP0709I NO IMAGE COPY OBTAINED FOLLOWING REORGANIZATION DBDNAME=`*`dbdname`* `DDNAME=`*`ddname`* message when condition found during Change Accumulation utility execution.

26

These changes cause IMS to issue new and different messages for some Change Accumulation utility failures.

The following message is no longer issued:

`DSP0711I SPECIFIED PURGETIME IS NOT THE CURRENT IMAGE COPY TIME DBDNAME=`*`dbdname`*
`DDNAME=`*`ddname`*

The DSP0715E message is issued by the Change Accumulation utility when the purge time specified on the control statement is later than the current purge time. The current purge time is the last Image Copy time. The utility does not process since it would produce unusable data. That is, data required for recovery would not be included in the output data set.

The DSP0717E message is issued by the Change Accumulation utility when there are updates following an offline reorganization without an Image Copy following the reorganization. The utility does not process since it would produce unusable data. Without an Image Copy following the reorganization, the Change Accumulation data could not be used. Previously, the DSP0709I message was used to indicate this condition.

Change Accumulation Enhancement (IMS 10 SPE)

- Changed message explanation, system action, and user response
  - `DSP0709I NO IMAGE COPY OBTAINED FOLLOWING REORGANIZATION DBDNAME=`*`dbdname`*` DDNAME=`*`ddname`*
    - The previous text in the *Messages and Codes Reference* for DFS0709I indicated that the message could be issued by the execution of the Change Accumulation utility, not just by the GENJCL.CA command execution.
    - The message is issued by GENJCL.CA after an offline reorganization
      - Image copies are not required following a HALDB Online Reorganizations (OLR)
        - OLR creates new data sets; therefore, the log records contain all of the data to recreate these data sets

27

The new text in the *Messages and Codes Reference* for DFS0709I is the following:

- Explanation
  - This message is received during the GENJCL.CA process when an offline reorganization of the identified database data set (DBDS) or area data set (ADS) was recorded in the RECON data set, but there is no record of a usable image copy data set created and no online reorganization since that offline reorganization.
- System action
  - The GENJCL.CA processing continues as long as there are other DBDSs to process but the identified DBDS is removed from the generated JCL job. If there are no other DBDSs to process, the GENJCL.CA command fails.
- User response
  - Take an image copy for the DBDS. The next GENJCL.CA command will use the new starting point for the next Change Accumulation utility.

DELETE.DB Performance Enhancement (IMS 10 SPE)

IMS 10 APAR: PK68143

- Reduced elapsed time for deleting DB record with many DBDS or AREA records
  - Applies to systems not using Parallel RECON Access (PRA)
  - Only one MUP (multiple update) record is used for the LOGALL record
    - Previously, many MUP records were written for LOGALL
      - MUP was updated when each ALLOC record was deleted
      - DEDB can have hundreds of AREAs
        - Each AREA can have many ALLOCs
  - No change to externals
    - Enhancement occurs without user action
- Benefits
  - Reduces elapsed time and RECON reserve time

28

DELETE.DB performance is enhanced by an SPE for IMS 10 (APAR PK68143 and PTF UK43479).

DELETE.DB processing can take a long time before this enhancement is implemented. The reason the processing of DELETE.DB takes so long is that all the processing is under one multiple-update scope. As a DBDS or AREA record is deleted, each of its ALLOC records is deleted. As the ALLOCs are deleted, the associated LOGALL record is updated. So, each time the LOGALL record is updated, a MUP record with the before image is written. If there is a large number of ALLOCs, the same LOGALL record can get updated several hundred (thousand) times. Each update would write a MUP record. Additionally, if a LOGALL record has a large number of entries, it is probably segmented, meaning there will be additional associated physical I/O. Since we are inserting thousands of MUPs all with a descending KEY, there will be many CI/CA splits.

The solution is to only write the MUP for the first CHANGE of a given record in a given MUP sequence. MUPs are meant to restore the RECON to the state it was at before any processing, so only the first before image is needed. All subsequent MUPs changes for that record are pointless.

This enhancement is made to DELETE.DB since it is the only command that typically changes the same record multiple times.

DBRC Enhancements

- BPE Based DBRC Region
  - Exit management, tracing, and simplified definition of LSR buffers
- Unconditional Deletion of PRILOG information
  - Simplified RECON maintenance
- Security override for non-production RECON copies
  - Simplified process for RECON problem investigation
- User Change Accumulation support
  - Support for IMS tools enhancements
- Change Accumulation enhancement (SPE for IMS 10)
  - RECON contents may change after the CA JCL is generated

DBRC Migration and Coexistence

Supported Migrations and Coexistence

- IMS 9 to IMS 11
  - Apply DBRC coexistence SPE APAR PK61582 to IMS 9
  - Upgrade RECONs from IMS 9 to IMS 11
- IMS 10 to IMS 11
  - Apply DBRC coexistence SPE APAR PK61583 to IMS 10
  - Upgrade RECONs from IMS 10 to IMS 11

31

IMS 9 RECONs may be upgraded directly to IMS 11. Similarly, IMS 10 RECONs may be upgraded to IMS 11. There is no support to upgrade RECONs from previous releases directly to IMS 11.

PK61582 is an IMS 9 SPE (Small Programming Enhancement) APAR. It allows IMS 9 to use RECONs which have been upgraded to IMS 11.

PK61583 is an IMS 10 SPE APAR. It allows IMS 10 to use RECONs which have been upgraded to IMS 11.

These APARs should be applied to IMS 9 or IMS 10 before its RECONs are upgraded to IMS 11.

RECON Listings

- "COEXISTENCE LEVEL" in subsystem record listing
  - Added by IMS 10 and IMS 10 coexistence SPE for IMS 9
  - May be used to determine if subsystems would cause an upgrade failure

```
SSYS
 SSID=IMS1      LOG START=08.067 17:25:44.2
 SSTYPE=ONLINE  ABNORMAL TERM=OFF  RECOVERY STARTED=NO   BACKUP=N
 TRACKED=NO     TRACKER TERM=OFF   SHARING COVERED DBS=NO
 IRLMID=**NULL**  IRLM STATUS=NORMAL        GSGNAME=**NULL**
 COEXISTENCE LEVEL=11.1

 AUTHORIZED DATA BASES/AREAS=4      VERSION=9.1   XRF CAPABLE=NO
                                                   ENCODED
   -DBD-        -AREA-    -LEVEL-  -ACCESS INTENT-  -STATE-
   PDHDOKA      **NULL**     0        UPDATE           6
   PDHDOKB      **NULL**     0        UPDATE           6
   PDHDOKC      **NULL**     0        UPDATE           6
   PDHDOKD      **NULL**     0        UPDATE           6
```

- In this example the subsystem is at 9.1 but has the 11.1 coexistence maintenance applied

32

IMS 10 added the coexistence level to the RECON listing of subsystem records. This was also added to IMS 8 and IMS 9 by the IMS 10 coexistence SPEs for these releases. The VERSION= field indicates the IMS release level of the subsystem. The COEXISTENCE LEVEL= field indicates if the coexistence maintenance for a later release has been applied. In this example, the IMS 11 DBRC coexistence maintenance has been applied to the IMS 9 system used by this subsystem. This listing could have been produced by an IMS 9 or IMS 10 DBRC utility with the IMS 11 coexistence SPE applied or it could have been produced by the IMS 11 DBRC utility.

RECON Upgrade

- RECONs are upgraded after IMS 11 is installed
  - Upgrade must use the IMS 11 DBRC utility (DSPURX00)
- Two RECONs and a spare must be available
- CHANGE.RECON UPGRADE
  - May be executed while subsystems are running
    - Upgrade fails if there is a subsystem record for an IMS 9 or IMS 10 subsystem without the DBRC coexistence SPE
      - Some utilities do not create subsystem records
        - They are not protected by the check for subsystem records
        - If they are running without the SPE, unpredictable results may occur
        - Examples: Change Accumulation, Log Archive, DSPURX00, HALDB Partition Definition Utility (PDU), some DBRC API applications
  - May be invoked using the DBRC API

33

RECONs are upgraded to IMS 11 by using the DBRC CHANGE.RECON UPGRADE command with the IMS 11 DBRC utility (DSPURX00).

The upgrade process requires that there are two active RECON data sets with an available spare. The upgrade process upgrades the records in COPY1 and then copies COPY1 to the spare.

The upgrade may be run while the RECONs are allocated to and being used by IMS 9 or IMS 10. Of course, these systems must be able to use IMS 11 RECONs. The upgrade checks the RECONs to ensure that any subsystems using the RECONs are capable of using IMS 11 RECONs. It does this by examining the SUBSYS records in the RECONs. Some IMS utilities do not create SUBSYS records. Thus, the upgrade cannot determine if they are running. Users must ensure that any IMS utility which is running at the time of the upgrade has the appropriate maintenance (PK61582 or PK61583) which allows it to read IMS 11 RECONs.

IMS 10 added the capability to issue DBRC commands from programs using the DBRC API. This includes the capability to issue the CHANGE.RECON UPGRADE command.

RECON Upgrade

- Some RECON records are larger in IMS 11
  - Examples:
    - RECON header extended by 44 bytes for 'RECON qualifier'
    - Change Accum execution record extended by 16 bytes
    - Database/Area Authorization records extended by 20 bytes
  - Upgrade may increase the size of the RECONs

- Recommendation
  - Ensure that RECONs have room for expanded records
    - May require availability of secondary extents

RECON Upgrade

- Upgrade processing from IMS 10 to IMS 11
  - Reads SSYS records to check for DBRC SPE
  - Updates RECON header record
    - Expands record by 44 bytes for 'RECON qualifier' field used by DBRC Security Override support
    - Sets version indicator and MINVERS value
  - Updates RECON header extension record
    - Sets version indicator
  - Updates Change Accumulation execution records
    - Expands records by 16 bytes
  - Updates Database/Area Authorization records
    - Expands records by 20 bytes to support future enhancements
  - After COPY1 is upgraded, it is copied to COPY2

35

The upgrade of the RECONs includes the reading of the subsystem (SSYS) records to ensure that these subsystems are running with the DBRC coexistence SPE. If not, the subsystem could not use the RECONs and the upgrade fails.

The update changes a few records in the RECONs.

The header record is expanded for support of DBRC Security Override for Copies of RECONs. The version indicator is set to 11 and the MINVERS value is set to '9.1' if it previously was '8.1'. The Cross DBRC Service Level ID (CDSLID) is set to the higher of the value in the RECONs before the upgrade and "1".

The version indicator in the RECON header extension record is set to 11.

The Change Accumulation Execution records are expanded for support of user data.

The Database records and Area Authorization records are expanded by 20 bytes to support future enhancements.

The upgrade is done by upgrading the records in COPY1 and then copying it to COPY2.

RECON Upgrade

- Parallel RECON Access processing
  - RECON activity is quiesced
  - RECONs are closed and reopened in LSR mode
  - Records are upgraded
  - COPY1 is copied to the spare
  - RECONs are reopened in PRA mode
  - Quiesce is ended

36

If Parallel RECON Access is in effect, there cannot be any shunted I/O when the upgrade begins. The process begins with a quiesce close and a check for shunted I/O. The RECONs 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 copying COPY1 to the spare. After the upgrade completes, the RECONs are reopened in PRA mode and the quiesce is ended.

RECON Upgrade

- Upgrade from IMS 9 to IMS 11 also does the following
  - Sets ACCESS to SERIAL and sets LIST to STATIC in the RECON header record
  - Moves PLEXNAME and adds GROUPID in the RECON header extension record
  - Updates Image Copy records
    - IC type field expanded and moved
  - Updates Subsystem records
    - API flag added

37

When the RECONs are upgraded from IMS 9 to IMS 11, the upgrade includes actions that are required due to changes implemented in IMS 10.

The support for Parallel RECON Access in IMS 10 added the ACCESS and LIST values to the RECON header record. Their default values are set.

The support for fuzzy user image copies in IMS 11 caused the Image Copy records to be expanded.

The support for subsystems in the DBRC API added the API flag in the Subsystem records.

MINVERS

- IMS 11 MINVERS valid values
  - '9.1', '10.1', and '11.1'

- Upgrade of RECONs
  - MINVERS('8.1') changed to MINVERS('9.1')
  - MINVERS('9.1') remains MINVERS('9.1')
  - MINVERS('10.1') remains MINVERS('10.1')

- MINVERS 11.1 is required for Database Quiesce

MINVERS is the parameter on the INIT.RECON and CHANGE.RECON commands which controls the minimum level of IMS which may use the RECONs. The minimum level of IMS which can use IMS 11 RECONs is IMS 9. If the previous MINVERS value was for '8.1', it is changed to '9.1' by the upgrade. Otherwise, upgrades do not change the MINVERS value.

MINVERS 11.1 is required for the use of the Database Quiesce function which is introduced in IMS 11.

IMS 9 to IMS 11 Migration Considerations

- IMSplex Name change and DBRC Group ID introduced in IMS 10
  - IMSplex name is specified with 5 characters (xxxxx)
    - IMS 9 stores 'CSLxxxxx' in the RECON header extension record
    - IMS 11 stores 'xxxxxyyy' in the RECON header extension record
      - 'yyy' is the DBRC Group ID
        - DBRC Group ID defaults to '001'
        - Upgrade uses the default
  - Example
    - IMS 9: IMSplex name is MYPLX
      - Contents in RECON Header Extension record: 'CSLMYPLX'
      - RECON Listing: `IMSPLEX=MYPLX`
    - IMS 11 after upgrade of RECONs with IMSplex name of MYPLX
      - Contents in RECON Header Extension record: 'MYPLX001'
      - RECON Listing: `IMSPLEX=MYPLX      GROUP ID=001`

39

This is information for those migrating directly from IMS 9 to IMS 11. These changes were introduced to DBRC in IMS 10.

The IMSplex name is optional. It is required for Automatic RECON Loss Notification and Parallel RECON Access. The IMSplex name is specified with up to 5 characters. It is specified either in the IMSPLEX= execution parameter or by the DBRC SCI Registration exit routine. When first specified, it is stored in the RECONs. IMS 9 stores the IMSplex name as 'CSLxxxxx' where 'xxxxx' is the value specified in the IMSPLEX= parameter or in the exit routine. When the RECONs are upgraded to V10, the value stored is 'xxxxxyyy' where 'yyy' is the DBRC Group ID. The upgrade sets the DBRC Group ID to '001' which is the default value.

IMS 9, 10, and 11 list only the 5 characters of the IMSplex name in listings of the RECON header. These listing include a line with IMSPLEX=xxxxx when an IMSplex name has been stored in the RECONs. If there is no value stored, the line includes IMSPLEX=**NONE**. IMS 11 listings also include the DBRC Group ID on this line. If there is no IMSplex name the Group ID is listed as GROUP=**NONE**. If there is an IMSplex name, the Group ID is listed as GROUP=yyy where yyy is the Group ID.

IMS 9 to IMS 11 Migration Considerations

- IMSplex and DBRC Sharing Group ID Specification
  - IMSplex users considerations
    - DBRC SCI Registration Exit routine (DSPSCIX0) can set the DBRC Sharing Group ID
      - Recommended
      - If DBRCGRP is specified in EXEC, it is passed to the exit routine
  - Migration and coexistence considerations
    - DBRC Sharing Group ID in RECONs is tolerated by IMS 9 systems
      - Exit routine is not passed the Group ID
      - Exit routine does not specify the Group ID
      - IMS 9 systems will process all ARLN notifications from its IMSplex group
      - ARLN notifications from IMS 9 systems will be sent to all members of the IMSplex, without regard to IMS 10 or IMS 11 DBRC Group IDs
        - Do not use multiple DBRC Sharing Group IDs in an IMSplex where IMS 9 systems are used

This is information for those migrating directly from IMS 9 to IMS 11. These changes were introduced to DBRC in IMS 10.

The DBRC SCI Registration exit routine (DSPSCIX0) may be used to specify the IMSplex name, as in previous releases, and the DBRC Group ID which was introduced in IMS 10. The use of the exit routine is recommended for users of IMSplex. It removes the requirement to specify IMSPLEX= for the execution of all IMS jobs which use DBRC. This includes batch jobs and utilities. With IMS 10 and IMS 11 the exit also may specify the DBRC Sharing Group ID. This removes the requirement to specify DBRCGRP= for IMS executions.

IMS 9 systems can tolerate the specification of the DBRC Group ID in the RECONs. DBRCGRP= is not a valid parameter on the EXEC statement for IMS9. When the exit routine is invoked in an IMS 9 environment, the DBRC Group ID is not passed to it. The exit routine cannot specify the DBRC Group ID. Even though an IMS 9 instance cannot specify the DBRC Group ID, it can join an IMSplex where IMS 10 or IMS 11 instances are using DBRC Group IDs. The IMS 9 instance will be passed all ARLN notifications from the IMSplex group. If an IMS 9 system reconfigures its RECONs, its ARLN notification will be processed by all members of the IMSplex. This will include all IMS 10 and IMS 11 systems. If there are multiple DBRC Groups, all members of all groups will process the notification. For these reasons, you should not use multiple DBRC Group IDs in an IMSplex while you are still using IMS 9 systems.

RECON Listings

- Planning considerations for migration from IMS 9 to IMS 11
  - IMS 10 added information to RECON status listing

```
RECON
 RECOVERY CONTROL DATA SET, IMS V11R1
 DMB#=231                           INIT TOKEN=06082F0536577F
 NOFORCER  LOG DSN CHECK=CHECK44    STARTNEW=NO
 TAPE UNIT=3480       DASD UNIT=SYSDA      TRACEOFF   SSID=**NULL**
 LIST DLOG=NO                 CA/IC/LOG DATA SETS CATALOGED=NO
 MINIMUM VERSION = 9.1        CROSS DBRC SERVICE LEVEL ID= 00001
 REORG NUMBER VERIFICATION=NO
 LOG RETENTION PERIOD=00.001 00:00:00.0
 COMMAND AUTH=NONE  HLQ=**NULL**
 RCNQUAL=**NULL**
 ACCESS=SERIAL       LIST=STATIC
 SIZALERT DSNUM=15        VOLNUM=16      PERCENT= 95
 LOGALERT DSNUM=3         VOLNUM=16

 TIME STAMP INFORMATION:

   TIMEZIN = %SYS

   OUTPUT FORMAT:  DEFAULT = LOCAL  NONE   PUNC YY
                   CURRENT = LOCAL  NONE   PUNC YY

 IMSPLEX = ** NONE **   GROUP ID = ** NONE **

-DDNAME-       -STATUS-       -DATA SET NAME-
 RECON1         COPY1          IMSTESTS.DSHR.RECON1
 RECON2         COPY2          IMSTESTS.DSHR.RECON2
 RECON3         SPARE          IMSTESTS.DSHR.RECON3
```

41

The RECON status or header listing has some added and changed information.

Of course, the IMS version is listed as "V11R1". This means that the RECONs have been upgraded to IMS 11.

IMS 11 introduces a new line for the RECON qualifier (RCNQUAL). RCNQUAL is used for determining if DBRC command authorization security override should be in effect for this set of RECONs. It is compared with the data set name for COPY1.

IMS 10 introduced a new line which lists the type of RECON access, either SERIAL or PARALLEL. On the same line the default for the DBRC LIST command, either STATIC or CONCURR, is shown.

On the line where the IMSPLEX value is shown, the DBRC Group ID value is also shown. In this example, these parameters have no values so "** NONE **" is listed. The DBRC Group ID was introduced by IMS 10.

The sample listing shown here includes the "CROSS DBRC SERVICE LEVEL ID". This also appears on IMS 9 RECON listings when the maintenance for APARs PQ98655 and PK01097 is applied. The service level ID is used to invoke functions which require a consistent level of maintenance on all IMS systems using the RECONs.

DBRC Migration Steps

1. Install IMS 9 or IMS 10 DBRC Migration/Coexistence SPEs
2. Install IMS 11 DBRC Type 4 SVC
   – The IMS 11 Type 4 SVC may be used with IMS 9 or IMS 10
3. Upgrade RECONs using the IMS 11 SDFSRESL library
   – Recommendation: CHANGE.RECON specifying RECON qualifier
4. Begin using IMS 11

5. Discontinue all use of IMS 9 and IMS 10
6. CHANGE.RECON MINVERS('11.1')
   – Full precision is used in timestamps with '11.1' or '10.1'
   – Full precision is not used in timestamps with '9.1'

This shows the DBRC steps for migration to IMS 11.

The first set of steps allows you to begin using IMS 11. The migration/coexistence SPE must be installed on the old release before you upgrade the RECONs to IMS 11. The IMS 11 DBRC Type 4 SVC must be installed before you may use IMS 11. The upgrade of the RECONs to IMS 11 requires that you use the SDFSRESL library created by the installation of IMS 11. The upgrade using this library will be to the IMS 11 format. Once the RECONs have been upgraded, you may begin using IMS 11. You may also continue to use IMS 9 or IMS 10. After you have upgraded the RECON, you may want to specify the RECON qualifier by issuing the CHANGE.RECON command.

Once you have discontinued all use of IMS 9 and IMS 10, you can change the MINVERS value to '11.1'. If you change MINVERS from '9.1' to '10.1' or '11.1' IMS will begin using the increased precision timestamp. Before changing MINVERS to '11.' or '10.1', you must ensure that the IMS utility control statements that you use specify full precision in their timestamps. The control statements generated by GENJCL statements will always generate control statements with the correct timestamps. Remember that the position of the timestamp in the control statements for the IMS 10 and IMS 11 Change Accumulation and Database Recovery utilities does not depend on the MINVERS value, however, if MINVERS is not '10.1' or '11.1' the low order part of the timestamp does not matter since these positions in timestamps are not recorded in the RECONs unless MINVERS('11.1') or MINVERS('10.1') is specified.

IMS Version 11

Installation and Migration

Information Management software

Installation and Migration

- Packaging, Prerequisites, and Coexistence
- IMS Library Changes
- IVP and Syntax Checker Enhancements
- Installation and Migration Tasks
- Review of Migration Considerations

Packaging, Prerequisites, and Coexistence

Packaging

- IMS 11 product number: 5635-A02

FMID	Feature Description
HMK1100	System Services
JMK1101	Database Manager
JMK1102	Transaction Manager
JMK1103	Extended Terminal Option (ETO)
JMK1104	Recovery Level Tracking (RSR)
JMK1105	Database Level Tracking (RSR)
JMK1106	IMS Java and On Demand
HIR2220	IRLM 2.2

4

IMS V11 packaging is the same as that for IMS V10 and IMS V9. Transaction Manager is a prerequisite for ETO. Recovery Level Tracking RSR is a prerequisite for Database Level Tracking RSR.

IRLM 2.2 is the only IRLM shipped and supported with IMS V11.

Software Prerequisites

- Minimum software level prerequisites
  - z/OS V1R9 (5694-A01)
    - RACF, or equivalent, if security is used
    - High Level Assembler Toolkit Release 5 (5696-234)
  - IRLM 2.2, if IRLM is used
- Minimum software levels for optional functions:
  - The DLIModel utility plug-in
    - Requires Eclipse version 3.2.2 and either Rational Developer for System z (RDZ), version 7.1.0 or Rational Application Developer (RAD), version 7.0.0.4
  - Java requires SDK 6.0
  - Parallel RECON Access (introduced in IMS 10) requires Transactional VSAM
    - Special bids (price discounts) are considered for Transactional VSAM
  - See the IMS 11 Release Planning publication for additional requirements

5

The minimum level of z/OS for IMS 11 is z/OS V1R9. In addition to z/OS the user must install RACF, or an equivalent security product, in order to use security with IMS 11. RACF is part of the SecureWay Security Server. As with previous IMS releases, the High Level Assembler Toolkit is required to provide assembler macros that IMS uses. If the IRLM is used, IRLM 2.2 is required. Program Isolation (PI) is also supported with IMS 11. IRLM is required for block level data sharing.

The DLIModel utility is an Eclipse plug-in. It requires Eclipse version 3.2.2 with either Rational Developer for System z (RDZ), version 7.1.0 or Rational Application Developer (RAD), version 7.0.0.4.

The use of Java with IMS 11 requires SDK 6.0.

Special bids will be considered for IMS customers using the Parallel Recon function, who do not already have DFSMStvs, to acquire DFSMStvs for use restricted to IMS.

The IMS 11 Release Planning publication has additional information about requirements when using particular functions in IMS 11. This is especially important for Java users. The level of JDK or SDK depends on the environment in which Java will execute.

z/OS 1.10 is not required when implementing the optional Fast Path 64 bit buffer manager, but it might be advisable to implement this function under z/OS 1.10. When running under z/OS 1.9 this buffer manager has additional overhead.

Hardware Prerequisites

- The hardware prerequisites are the same as for IMS 10
  - IMS 11 runs only on 64 bit processors running in z/Architecture mode
- Sysplex data sharing requires Coupling Facility level 9 or later
- Shared queues and shared EMH require coupling facility level 9 or later
  - System-managed CF duplexing requires CF level 12

6

These are the same processor and Coupling Facility levels that are required for IMS 10.

Supported Connections

- ISC is supported with
  - IMS 11, IMS 10, and IMS 9
  - CICS Transaction Server V3.2 and V3.1
  - User-written software
- MSC is supported with
  - IMS 11
  - IMS 10
  - IMS 9
- Shared Queues is supported with
  - IMS 11
  - IMS 10
  - IMS 9

7

All currently supported releases of IMS and CICS are supported for ISC connectivity to IMS 11.

All currently supported releases of IMS are supported for MSC connectivity to IMS 11.

All currently supported releases of IMS are supported for shared queues with IMS 11 Transaction Manager.

Supported Connections

- DB2 connections are supported with
  - DB2 for z/OS 9 and 8
- DBCTL connections are supported with
  - CICS Transaction Server V3.2 and V3.1
- IRLM 2.2 connections to IRLM 2.1 are supported
  - IRLM 2.1 is supported with IMS V9
  - IRLM 2.1 is not supported with IMS V11 or IMS V10
  - IRLM 2.2 with IMS V11 or IMS V10 may connect to IRLM 2.1 with IMS V9

8

All currently supported releases of DB2 on z/OS are supported for connections from IMS 11.

All currently supported releases of CICS are supported for DBCTL connectivity to IMS 11.

Supported Migrations and Coexistence - DBRC

- IMS 10 to IMS 11
  - Upgrade RECONs from IMS 10 to IMS 11
    - IMS 10 SPE PK61583 allows IMS 10 to use IMS 11 RECONs
- IMS 9 to IMS 11
  - Upgrade RECONs from IMS 9 to IMS 11
    - IMS 9 SPE PK61582 allows IMS 9 to use IMS 11 RECONs

9

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. Before doing the upgrade you should apply the Small Programming Enhancement to your IMS 9 or IMS 10. This allows the IMS 9 or IMS 10 systems to use the RECONs after they have been upgraded to IMS 11.

Migration with DRD

- IMS 11 system may be cold started with RDDS from IMS 10
  - IMSID must remain the same
- IMS 10 system may be cold started with RDDS from IMS 11 for fall back
  - IMSID must remain the same

Installations which have implemented DRD in IMS 10 may carry their DRD definitions forward to IMS 11 by using the RDDS. When IMS 11 is cold started, it may import its database, program, transaction, and routing code definitions from an RDDS created with IMS 10.

Similarly, for a fall back from IMS 11 to IMS 10 you may cold start the IMS 10 system and import definitions from an RDDS created by IMS 11.

For both migration and fall back, the IMSID must be the same as that used by the system which created the RDDS.

CSL Coexistence

- IMS 11 and IMS 10
  - CSL address spaces and IMS may be at mixed levels
    - SCI, OM, RM, and IMS subsystems may be at both IMS 10 and IMS 11
  - IMS 11 is recommended for the CSL address spaces

- IMS 11 and IMS 9
  - IMS 9 has some coexistence SPEs for IMS 10 and IMS 11
    - These are documented on a later page in this class
  - With these coexistence SPEs, CSL and IMS may be at mixed levels
    - SCI, OM, RM and IMS subsystems may be at IMS 11, IMS 10, and IMS 9
  - IMS 11 is recommended for the CSL address spaces

11

When migrating a system using the Common Service Layer address spaces, any address space may be migrated to IMS 11 before the other address spaces are migrated. It is permissible to have some SCI, RM, OM, and IMS subsystems on IMS 11 while others are on IMS 10. There are some restrictions with IMS 9. IMS 9 has some coexistence SPEs that allow a mixture of IMS 9 and IMS 10 address spaces. These also apply to mixing IMS 9 and IMS 11 address spaces. With these SPEs applied to IMS 9, any mixture of IMS 9, IMS 10, and IMS 11 CSL address spaces and IMS subsystems is allowed.

In all cases, the use of IMS 11 CSL address spaces is recommended when any IMS subsystem in at IMS 11.

IMS Version 11

Coexistence Maintenance

- Open Database enhancements SPEs
  - Only required for using the ODBA interface delivered with IMS 11
    - IMS 10: PK66022
    - IMS 9: PK66020
      - These APARs add the CIMS CONNECT call to ODBA so that ODBA applications can connect to multiple IMS subsystems with a single call.

12

To use the Open Database enhancements in conjunction with an IMS Version 9 or IMS Version 10 system, you must apply the coexistence APARs. If you are using either IMS Version 9 or IMS Version 10, you must apply the appropriate coexistence APARs. These APARs add the CIMS CONNECT call to ODBA so that ODBA applications can connect to multiple IMS subsystems with a single call.

If you want your IMS Version 9 or IMS Version 10 ODBA application to use the ODBA interface delivered with IMS Version 11, you must:

1. Apply the appropriate coexistence APAR.
2. Modify your existing ODBA application servers to use ODBM by adding the IMSPLEX and ODBMNAME parameters to the DFSPRP macro.
3. Recompile and rebind the DFSPZPxxx load module.

After performing these tasks, you have the option of simplifying the ODBA applications by replacing multiple CIMS INIT commands with a single CIMS CONNECT command.

Other Coexistence Maintenance for IMS 9

- Global Online Change Coexistence APARs
  - IMS 9 - PK23402
- System Management Enhancement Coexistence SPEs
  - IMS 9 - PK30189
- Resource Consistency Checking Coexistence SPEs
  - IMS 9 - PK32970
  - Resource Consistency Checking is not done in IMS 11 or IMS 10
- Operations Management Coexistence SPEs
  - IMS 9 - PK27280

13

This chart shows the coexistence maintenance required on IMS 9 when it coexists with IMS 11.

The Global Online Change coexistence SPE (APAR PK23402; PTF UK20811) allows lower level IMS systems to share an OLCSTAT data set with IMS 11 or IMS 10. IMS 10 made changes to the header record. The SPE allows IMS 9 to implement and tolerate these changes.

The System Management Enhancement coexistence SPE (APAR PK30189; PTF UK22059) allows IMS 9 systems to coexist with IMS 11 or IMS 10. IMS 9 systems which process transaction submitted from the OM API will receive an AD status code if they reply to the IOPCB.

IMS 11 and IMS 10 do not support the resource consistency checking function of the Resource Manager. The Resource Consistency Checking SPE (APAR PK32970; PTF UK24486) allows IMS 9 systems to use this function among themselves while in an IMSplex with IMS 11 or IMS 10 systems.

The Operations Management coexistence SPE (APAR PK27280; PTF UK18913) allows OM address spaces and IMSs at IMS 9 to coexist in a CSL environment with IMS 11 or IMS 10 OM.

Coexistence - IMS Utilities

- Batch Backout, Log Archive, and Log Recovery
  - Use the utility from the IMS release that created the log
- IMS 11 Database Recovery utility
  - Accepts Image Copies produced by IMS 11, IMS 10, and IMS 9
  - Accepts HISAM Unloads produced by IMS 11, IMS 10, and IMS 9
  - Accepts Change Accum data sets produced by IMS 11, IMS 10, and IMS 9
  - Accepts logs produced by IMS 11, IMS 10, and IMS 9
- IMS 11 Change Accumulation utility
  - Accepts logs produced by IMS 11, IMS 10, and IMS 9
  - Accepts Change Accum data sets produced by IMS 11, IMS 10, and IMS 9

14

The Batch Backout, Log Archive, and Log Recovery utilities access one log. The release level of the utility must match the IMS release that was used to create the log.

The IMS 11 Database Recovery utility (DFSURDB0) accepts Image Copies, HISAM Unload data sets, Change Accumulation data sets, and IMS logs as inputs. These inputs may be created by any currently supported release of IMS.

The Change Accumulation utility accepts IMS logs and Change Accumulation data sets as inputs. These inputs may be created by any currently supported release of IMS.

DLI Model Utility

- The batch DLI Model Utility is not supported with IMS 11
  - GUI interface is supported

IMS 10 was the last version of IMS that supported the batch DLI Model Utility.

IMS Version 11

Remote Site Recovery (RSR) Migration/Coexistence

- IMS 11 RSR tracking system can process logs created by IMS 11, IMS 10, or IMS 9
- IMS 11 RSR Isolated Log Sender can send logs created by IMS 11, IMS 10, or IMS 9
- Logs created by IMS 11 cannot be processed by IMS 10 or IMS 9 tracking system or Isolated Log Sender

- Migration steps
  – Upgrade the RSR tracking system RECONs to IMS 11
  – Migrate RSR tracking system to IMS 11
  – Upgrade the active system RECONs to IMS 11
  – Migrate active system Transport Manager Subsystem (TMS) running Isolated Log Sender to IMS 11
  – Migrate active IMS to IMS 11

16

The migration of systems using 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. On the other hand, IMS 10 and IMS 9 tracking systems cannot accept logs produced by IMS 11 and the IMS 10 and IMS 9 ILSs cannot accept logs produced by IMS 11. Of course, you could migrate all of the RSR components at the same time. You would more likely prefer to migrate them in stages. The restrictions mentioned above imply that the order of migration of the components is as shown on the slide. 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.

Log Records

- Some log records have changed
- New log records have been added
- DSECTS for IMS log records may be generated by assembling:
  – ILOGREC RECID=ALL

17

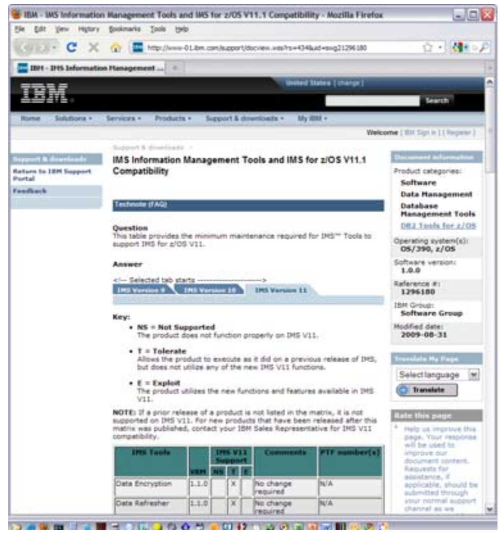
If you have application programs which process IMS log records, you should examine to see if they are affected by the changes to the log records. You can assemble DSECTs for IMS log records by using the ILOGREC macro.

Log records which are new in IMS 11 include the following:
X’221B’ CREATE, UPDATE, and DELETE commands for OTMADEST
X’4035’ checkpoint record for OTMA descriptors
X’4081’ checkpoint record for 64 bit Fast Path buffers
X’4515’ statistics from the 64 bit storage manager
X’5960’ FP 64 bit subpool creation, expansion, or deletion

Other log records are changed to support IMS 11.

IMS Tools Migration/Coexistence

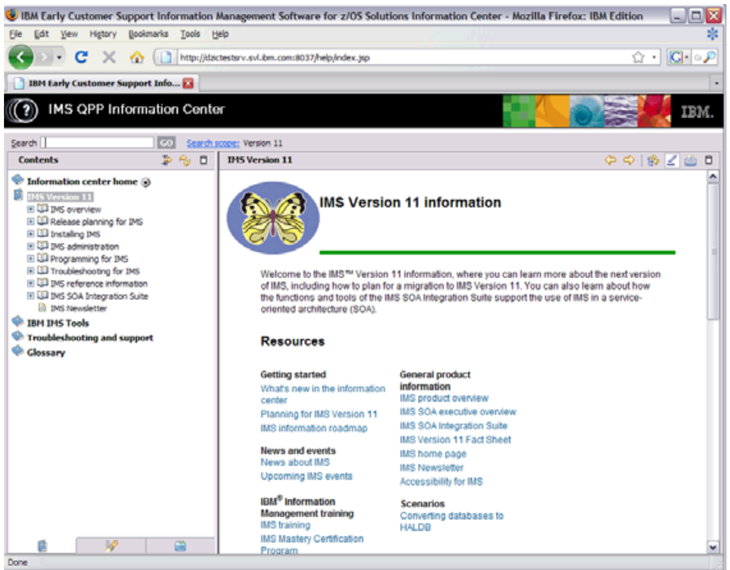
- Some products may require updates
  - Contact your vendor for information on requirements
  - IBM has a web site with consolidated information about requirements for IBM IMS tools with IMS 11
    - tinyurl.com/IMSToolsWithIMS11



IMS Library Reorganization

IMS Library

- Information Center contains information on IMS 11



20

The Information Center has been updated to include information on IMS 11.

IMS Library Reorganization

- IMS publication title changes and some consolidation
  – Chapter 5 of Release Planning explains the changes and organization
- *Optimized for viewing in the information center and PDF versions*
  – *BookManager format is NOT provided*

- *IMS 11 versus IMS 10 publications:*

IMS Version 11 Titles	IMS Version 10 Titles
Release Planning	*Release Planning Guide*
Installation	*Installation Verification Guide*
System Definition	*System Definition Guide* *System Definition Reference*
Exit Routines	*Exit Routine Reference*

21

IMS 11 includes some changes to the IMS publications. Many of the titles have been simplified. They now match the navigation tree titles in the Information Center.

IMS Library Reorganization

- IMS 11 versus IMS 10 publications:

IMS Version 11 Titles	IMS Version 10 Titles
Application Programming	Application Programming Planning Guide Application Programming Guide
Application Programming APIs	Application Programming API Reference
System Programming APIs	System Programming API Reference
Database Administration	Database Administration Guide
System Administration	System Administration Guide IMSplex Administration Guide
Communications and Connections	Communications and Connections Guide
Operations and Automation	Operations and Automation Guide

IMS Library Reorganization

- IMS 11 versus IMS 10 publications:

IMS Version 11 Titles	IMS Version 10 Titles
Command Reference, Volume 1: IMS Commands A-M	*Command Reference, Volume 1*
Command Reference, Volume 2 : IMS Commands N-Z	*Command Reference, Volume 2*
Command Reference, Volume 3: IMS Component and z/OS Commands	*Command Reference, Volume 3*
Database Utilities	*Database Utilities Reference*
System Utilities	*System Utilities Reference*

IMS Library Reorganization

- *IMS 11 versus IMS 10 publications:*

IMS Version 11 Titles	IMS Version 10 Titles
Diagnosis	*Diagnosis Guide* *Diagnosis Reference*
Messages and Codes, Volume 1: DFS Messages	*Messages and Codes Reference, Volume 1: DFS Messages*
Messages and Codes, Volume 2: Non-DFS Messages	*Messages and Codes Reference, Volume 2: Non-DFS Messages*
Messages and Codes, Volume 3: IMS Abend Codes	*Messages and Codes Reference, Volume 3: IMS Abend Codes*
Messages and Codes, Volume 4: IMS Component Codes	*Messages and Codes Reference, Volume 4: IMS Component Codes*

IVP and Syntax Checker Enhancements

IVP Enhancements

- New IVPs
  - Open Database (OPDB)
  - Callout application (COUT)
- Help Text Cleanup
  - Some new and more complete help

26

The IVP support of call out is application programs which invoke both synchronous and asynchronous callout.

Syntax Checker Enhancements

- Syntax Checker supports PROCLIB members for IMS 11, IMS 10, and IMS 9
  - IMS 8 members are not supported
- All previously supported members are supported
  - Including new, changed, or obsolete keywords
- New members supported
  - CSLDIxxx - ODBM initialization member
  - CSLDCxxx - ODBM configuration member
  - DSPBIxxx - DBRC initialization member
- IMS 11 Syntax Checker expands selected sections at start time
  - User does not have to "Expand all" under the View action bar

27

The Syntax Checker in IMS 11 supports PROCLIB members for IMS 11, IMS 10, and IMS 9. Of course, it can upgrade these members from IMS 9 or IMS 10 to IMS 11. The support includes all the keywords in the PROCLIB members. Some of these keywords may be new or changed and some may have become obsolete.

In IMS 10, the Syntax Checker supported the following members:

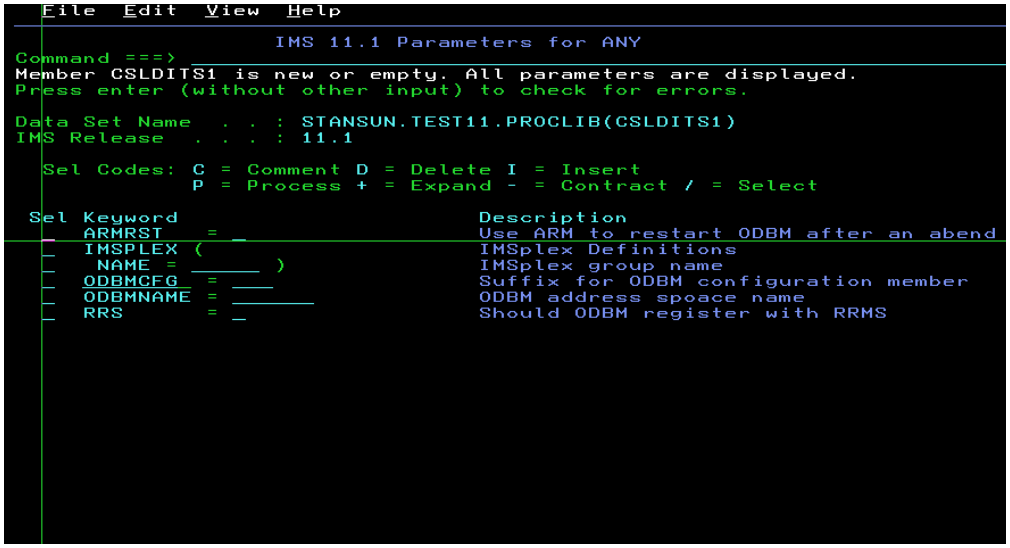
DFSPBxxx DFSDCxxx
DFSSQxxx – Shared Queues　　DFSDFxxx – Dynamic Resource Definition
DFSCGxxx – Common Service Layer　　CSLOIxxx – OM Initialization
CSLRIxxx – RM InitializationCSLSIxxx – SCI Initialization
CQSIPxxx – CQS Initialization　　CQSSLxxx – CQS Local Structure Definition
CQSSGxxx – Global Structure Definition　　BPE User Exit List member
IMS Connect Configuration member

IMS 11 adds support for:

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

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 action bar to see all of the data. IMS 11 changes this. In IMS 11 automatically expands selected sections at start time.

Syntax Checker - ODBM Initialization Member



This is the Syntax Checker panel used for the ODBM initialization member (CSLDIxxx).

Syntax Checker - ODBM Configuration Member

```
  File  Edit  View  Help
                    IMS 11.1 Parameters for ANY
Command ===> ____________________________________________________________
Member CSLDCTS1 is new or empty. All parameters are displayed.
Press enter (without other input) to check for errors.

Data Set Name  . . : STANSUN.TEST11.PROCLIB(CSLDCTS1)
IMS Release   . . . : 11.1

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

 Sel Keyword                          Description
  _   <SECTION=GLOBAL_DATASTORE_CONFIGURATION>    GLOBAL DATASTORE CONFIGURATION
  _    CNBA = ____                    Total number of Fast Path NBA buffers
  _    FPBOF = ___                    Fast Path DEDB overflow buffers number
  _    FPBUF = ___                    Fast Path DEDB buffers number
  _    IDRETRY = ___                  Number of times to attempt connection
  _    MAXTHRDS = ___                 Maximum concurrent active threads
  _    TIMER = __                     Time in seconds between attempts
  _   <SECTION=LOCAL_DATASTORE_CONFIGURATION>    LOCAL DATASTORE CONFIGURATION
  _    ODBM (                         Individual ODBM parameters list
  _     DATASTORE (                   IMS datastore characteristics
  _      ALIAS (                      Specifies an ALIAS value or values
  _       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 = ____ )                IMS datastore name
  _     NAME = ______ )               ODBM name
```


This is the Syntax Checker panel used for the ODBM configuration member (CSLDCxxx).

Syntax Checker - DBRC Initialization Member

```
  File  Edit  View  Help
                   IMS 11.1 Parameters for ANY
Command ===> ____________________________________________________
Member DSPBITS1 is new or empty. All parameters are displayed.
Press enter (without other input) to check for errors.

Data Set Name  . . : STANSUN.TEST11.PROCLIB(DSPBITS1)
IMS Release  . . . : 11.1

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

 Sel Keyword                            Description
  _  DBRCGRP  = ___                     Three-character DBRC group id
  _  IMSPLEX (                          IMSplex Definitions
  _    NAME = ___)                      IMSplex group name
  _  VSAMBUFF (                         VSAM LSR pool
  _    DATA = _____                     The number of data buffers
  _    INDEX = _____ )                  The number of index buffers
```


The is the Syntax Checker panel used for the DBRC Initialization member (DSPBIxxx).

Installation and Migration Tasks

IMS Version 11

Installation and Migration Tasks

- Migration Tasks
  - Review the IMS 11 *Release Planning* publication
  - Check PSP bucket
    - PSP upgrade name is IMS1110
  - Review the Program Directory
    - Available through the Info Center
  - Review the installation information in Chapter 1 of the *Installation* publication
  - Install prerequisite software and maintenance
    - Check your IMS tools and related products
  - Apply coexistence maintenance to other IMS systems

32

This is an overview of the tasks for migration to IMS 11.

As for all installations of new products the Preventive Service Planning (PSP) bucket and the Program Directory for the product should be reviewed before beginning the migration.

You should read the IMS 11 *Installation* publication before beginning the migration process. Chapters 1 should be reviewed for installation information.

Other products may need to be upgraded for use with IMS 11. They could require maintenance or new releases.

You should apply DBRC coexistence SPEs to your IMS 10 or IMS 9 systems before upgrading your RECONs to IMS 11. This is required for the IMS 10 or IMS 9 systems to be able to use the RECONs after the upgrade.

Similarly, you should apply the other coexistence SPEs to your lower level IMS systems.

Installation and Migration Tasks

- Migration Tasks (continued)
  - Evaluate and update IMS exit routines
    - RECON I/O Exit Routine (DSPCEXT0)
    - DFSMSCE0 must be reassembled
    - All IMS Connect exits must be reassembled when migrating IMS Connect
    - HWSIMSO0 and HWSIMSO1 are not shipped with IMS 11
  - Install IMS 11 using SMP/E installation process
    - CBPDO or ServerPac may be used
  - System definition

33

The only user exit routine that must be updated for use with IMS 11 is the RECON I/O Exit Routine (DSPCEXT0). If you use a RECON I/O Exit Routine, you should examine it for required changes due to 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 shipped 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.

CBPDO is Custom-Built Product Delivery Offering. 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 an entitled 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

System definition is required as with previous IMS releases. Most system definition statements from previous IMS releases are compatible with IMS 11.

Installation and Migration Tasks

- Migration Tasks (continued)
  - Install the Type 2 and Type 4 SVCs
  - Upgrade RECONs
    - Specify RECON qualifier after the upgrade
  - ACBGEN
  - Run the IVP

34

System definition creates the Type 2 and Type 4 SVC modules which must be installed in the z/OS system. A z/OS IPL is not required. They may be installed by running DFSUSVC0 and specifying SVCTYPE=(2,4).

Upgrade the RECONs by using the CHANGE.RECON UPGRADE command using the IMS 11 release of the DBRC utility. Specifying the RECON qualifier is not required, but it is recommended.

An ACBGEN is required for use with the online system or any batch DBB jobs.

Running the IVP is optional, but recommended. All required installations tasks are done outside of the IVP. The IVP verifies that the installation is correct.

Review of Migration Considerations

Dynamic Abend Dump Exit

- IMS 11 automatically installs new dump format exit routine (DFSAFMX0)
  - Previous versions of IMS required manual installation of the IMS Dump Formatting routine (DFSAFMD0)
    - Bind DFSAFMD0 into SYS1.LPALIB or an MLPA library
    - Add DFSAFMD0 name to IEAVADFM CSECT of IGC08054A in SYS1.LPALIB
  - IMS 11 dynamically adds the new module (DFSAFMX0)
    - DFSAFMD0 is not used by IMS 11
- Migration consideration
  - Do not delete DFSAFMD0 from system while IMS 10 or previous versions of IMS are in use
- Benefit
  - Simplifies IMS installation process

IMS 11 eliminates the need to manually install the IMS dump formatting routine (DFSAFMD0). IMS 11 dynamically installs a new module, DFSAFMX0. This eliminates the requirement to add DFSAFMD0 to the z/OS system when installing a new IMS release.

With previous versions of IMS, a bind of DFSAFMD0 into SYS1.LPALIB or an MLPA library was required and DFSAFMD0 had to be added to the IEAVADFM CSECT of IGC0805A in SYS1.LPALIB. Each new version of IMS had a new version of the DFSAFMD0 module. These modules supported the release with which they were shipped and previous IMS releases. IMS 10 and previous releases still requires DFSAFMD0. IMS 11 does not require this module.

Migration consideration:

IMS 10 and earlier versions of IMS still require the DFSAFMD0 module. It should not be deleted from the system until IMS 10 or earlier versions are no longer used in the system.

RACF Mixed-case Password Support

- IMS 11 has changed the default values for PSWDC and PSWDMC
  - Defaults are PSWDC=R and PSWDMC=R
    - Defaults indicate that RACF specifications are to be used for mixed-case
      - PSWDC specifies IMS option
      - PSWDMC specifies IMS Connect option
    - IMS online system does not have to be restarted when RACF changes from uppercase to mixed
      - IMS dynamically changes
  - IMS 10 defaults are PSWDC=U (uppercase) and PSWDMC=N (not mixed-case)
- IMS 11 default is appropriate for all users

37

The default values for PSWDC and PSWDMC are different for IMS Version 11 than they were in IMS Version 10. The new default values are PSWDC=R and PSWDMC=R (use the RACF specification).

IMS Version 9 is not capable of processing mixed-case passwords. IMS Version 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 Version 10 are PSWDC=U (uppercase) and PSWDMC=N (not mixed-case).

The PSWDC and PSWDMC parameters are enhanced in IMS Version 11 with the 'R' specification, which means that IMS and IMS Connect should handle passwords in the same manner as is specified in RACF.

IMS 11 changes the size of the OTMA control block DFSYTIB by 64 bytes (x'40') from the old size of x'110' to the new size of x'150'. Correspondingly the YTIB macro has been changed to recognize the new size. Any user modifications or non-IBM vendor software which references this block must be changed and/or re-assembled to recognize the new size.

IMS Connect Migration Considerations

- HWSCFGx
  - Single SSLPORT restriction
    - Ensure that the SSLPORT parameter only specified one port
    - Previous releases allow initialization to complete with multiple SSL ports
      - IMS 11 produces U3401 abend at initialization
  - IMS Connect display output has new information and fields that reflect the enhancements discussed in this section
    - Modify automation programs and MTO documentation as needed
- New/changed messages that affect automation and MTO documentation
  - HWSP1410W enhancement for errors occurring during storage release
  - HWSX0908W - issued if HWSIMSO0/HWSIMSO1 continue to be specified
  - HWSS0772W and HWSS0773I – issued in support of WARNSOC/WARNINC

The next two visuals document the migration considerations for IMS Connect.

IMS Connect environments, in previous releases, 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. In IMS 11, IMS Connect initialization will fail if multiple SSL ports are specified. The HWSCFGx member needs to be modified to only specify one port.

Several new specifications in the HWS, TCPIP and DATASTORE statements of the HWSCFGx configuration file are added to the display output of commands such as VIEWHWS, VIEWPORT, VIEWDS, etc. Automation programs and MTO documentation should be modified to recognize these new fields.

Automation programs that read the output of IMS Connect displays or query the HWSP1410W message need to be aware of the new information and fields that have been added by the IMS 11 enhancements. Similarly, Master Terminal Operators (MTOs) that issue IMS Connect commands should understand that additional information is provided.

Message HWSX0908W is issued if the old exits HWSIMSO0/HWSIMSO1 continue to be specified in the IMS Connect configuration member.

If WARNSOC and WARNINC are specified in the TCPIP HWSCFGx statement then new messages will be issued when the warning level is reached (HWSS0772W) and when the number of sockets falls below the warning level (HWS0773I).

IMS Connect Migration Considerations

- Required re-assembly of IMS Connect user message exits
  - Incorporates the expansion of the HWSEXPRM macro and XIBDS
- TCP/IP Automatic Reconnect
  - Remove directions to issue OPENPORT command from MTO documentation or automation for situations when the network bounces while IMS Connect is active
- Generated Client ID
  - Replace the resource adapter with the new one to take advantage of the support
- Recorder Trace
  - Old tracing remains in effect until the HWSRCORD DD is removed
    - BPE external trace datasets may contain new variable length trace entries as well as fixed length trace entry data

40

Because the HWSEXPRM macro has been expanded, IMS Connect exit routines that invoke the macro must be re-assembled. As a reminder, from the OTMA Resource Monitoring section, the XIBDS (Exit Interface Block Data Store Entry) has also been expanded.

Operator commands that are issued to ensure that IMS Connect reestablishes connectivity with a TCP/IP network are no longer needed. 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.

To take advantage of the Generated Client ID function, the IMS TM resource adapter must be replaced with the new version.

The new Recorder 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. One the new function has been enabled, new RCTR entries in the BPE external trace datasets will be introduced as variable length trace entries.

ODBA and ODBM Coexistence

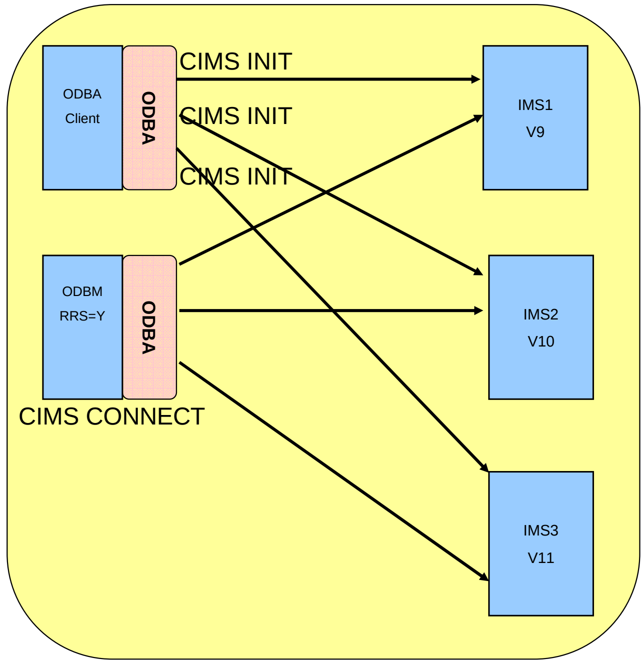
New CONNECT subfunction

Call AERTDLI parmcount, CIMS, AIB

parmcount = set to n (optional)

AIB = Address of AIB

- AIBSFUNC=CONNECT (required)
- AIBRSA1 = A(Parm list) (required)



Coexistence APARS

For IMS Version 9: PK66020
For IMS Version 10: PK66022

41

The ODBA interface from previous versions of IMS can coexist with IMS Version 11 without modification.

The V9/V10 ODBA coexistence apars are required to allow an ODBM to connect to a V9/V10 IMS subsystem. ODBM uses the new ODBA CIMS CONNECT support. There is supporting code that is necessary in the IMS core ODBA modules that is required in a V9/V10 IMS system in order to allow a V11 ODBM to communicate with that V9/V10 IMS subsystem thru ODBA.

The CIMS CONNECT call can be used by existing or new ODBA application programs.

The CIMS INIT call requires a separate call to each IMS.

Fast Path 64 Bit Buffer Manager

- Fast Path buffers above the 2GB bar in control region address space
  - Optional
  - DFSDFxxx PROCLIB member

```
<SECTION=FASTPATH>
FPBP64=Y,FPBP64M=xxxxxxxxxx
```

- Migration consideration
  - FDBR and XRF systems must have the same specification as the system which they are tracking.

Fast Database Recovery (FDBR) and XRF systems must have the same specifications in the FASTPATH section of their DFSDFxxx PROCLIB member that the systems they are tracking have.

DBRC Migration Recommendation

- Specify RECON qualifier when migrating to IMS 11

```
CHANGE.RECON ... CMDAUTH(SAF|EXIT|BOTH|NONE,safhlq,rcnqual)
```

  – Simplifies the handling of security for copies of RECONs
    - Especially important when shipping copies outside your installation
      – Others do not have to be given security information to investigate a problem

  – Specifying the qualifier during migration avoids the need to change it as part of problem investigation process

When you migrate to IMS 11 is a good time to specify the RECON qualifier. This avoids the need to change it later when you are doing problem determination. The RECON qualifier is used to determine is DBRC command authorization security should be enforced. It is only enforced when the RECON data set name for COPY1 includes the qualifier.

Other Sources of IMS 11 Information

- White Paper
  - IMS Version 11 Installation and Migration Tips
    - http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101605
- Presentation
  - IMS 11 - Easier than Ever - A Presentation
    - http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS3871