System Enhancements

Information Management software

System Enhancements …

- Dynamic resource definition (DRD) enhancements
- zAAP/zIIP times in accounting log records
- zIIP support for some CQS processing
- Member online change NAMEONLY option
- EAV support for non-VSAM data sets
- IMS logging enhancements
  - Extended-format data set support for OLDS/SLDS
  - IMS log buffers in 64-bit virtual storage
  - Simplified WADS management
- System pools in 64-bit real storage
- Sysplex Serial Program Management (SSPM) Disabling
- SPE for log archiving of non-recoverable databases

IMS 12 includes several system enhancements that will be discussed in the topic.

System Enhancements

- Command enhancements
- System usability enhancements
  - IMS 11 SPE User Exit Enhancements
- RAS enhancements
  - CQS traceability enhancements
  - IMS Dump Formatter enhancements
  - Reduce IMS Module Aliases in RESLIB
  - End-Of-Memory / End-Of-Task (EOT) Tracing Facility
  - IMS 11 SPE BPE Trace Selective Initialization
  - /DIAGNOSE SNAP command enhancements
- MIPS reduction enhancements

Dynamic Resource Definition (DRD) Enhancements

- New UPDATE option for IMPORT command
  - Previously, IMPORT could only be used for adding runtime resource definitions/descriptors that did not exist in the target IMS system
  - New IMPORT .. OPTION(UPDATE) allows existing runtime resource definitions/descriptors in the the target IMS to be changed
    - Command fails if changed definition is in use
- DRD usage of the IMS repository function
  - Previously, stored resource definitions/descriptors were kept in resource definition data sets (RDDSs)
  - New IMS repository function provides an additional method for storing stored resource definitions/descriptors
  - IMS repository will be described in a later topic
- Benefits
  - Improved manageability for DRD

10

For Dynamic Resource Definition (DRD) there are two enhancements in IMS 12.

Existing runtime resource definitions and descriptors can now be updated with the new OPTION(UPDATE) parameter on the IMPORT command. Previously with the IMPORT command, only new runtime resource definitions and descriptors could be added; if an existing definition was encountered, it was ignored.

IMS 12 provides another option for storing resource definitions and descriptors called the IMS Repository. Previously the only option for storing definitions was by using RDDSs (resource definition data sets).

zAAP/zIIP Times in Accounting Log Records

- zAAP/zIIP time field is added to the x'07', x'0A07' and x'56FA' log records
  - X'07' program termination
  - X'0A07' CPIC program termination
  - X'56FA' optional transaction level statistics record
- CPU time field is changed to include only the standard CP (central processor) time, not zAAP/zIIP time
  - Sum of CP and zAAP/zIIP times is the total CPU time
- Benefits
  - Users can distinguish between CP and zAAP/zIIP times
    - Could be used for accounting or charge out purposes
    - Significant for software licensing
    - Most significant for JMP and JBP regions

11

This enhancement was added to IMS 12 by APAR PM36273.

Log records with CPU time accounting data have been modified to distinguish between CPU time used by zAAP and zIIP processors versus time used by CPs (central processors). The x'07', x'0A07' and x'56FA' log records now have separate fields for zAAP and/or zIIP CPU time. The CPU time used by zAAP or zIIP processors is no longer included in the CPU time field. The CPU time field includes only time used by CPs.

Some servers are configured with zAAP or zIIP processors which run at a faster speed than the normal CP processors. In this case zAAP time and zIIP time is normalized to the equivalent time it would take to run on a normal CP when accumulated into the total CPU time. zAAP or zIIP processors run at a faster speed when CP processors are "capped" to limit the amount of processing that they may do. This is typically done to reduce the charges for the use of the CP processors.

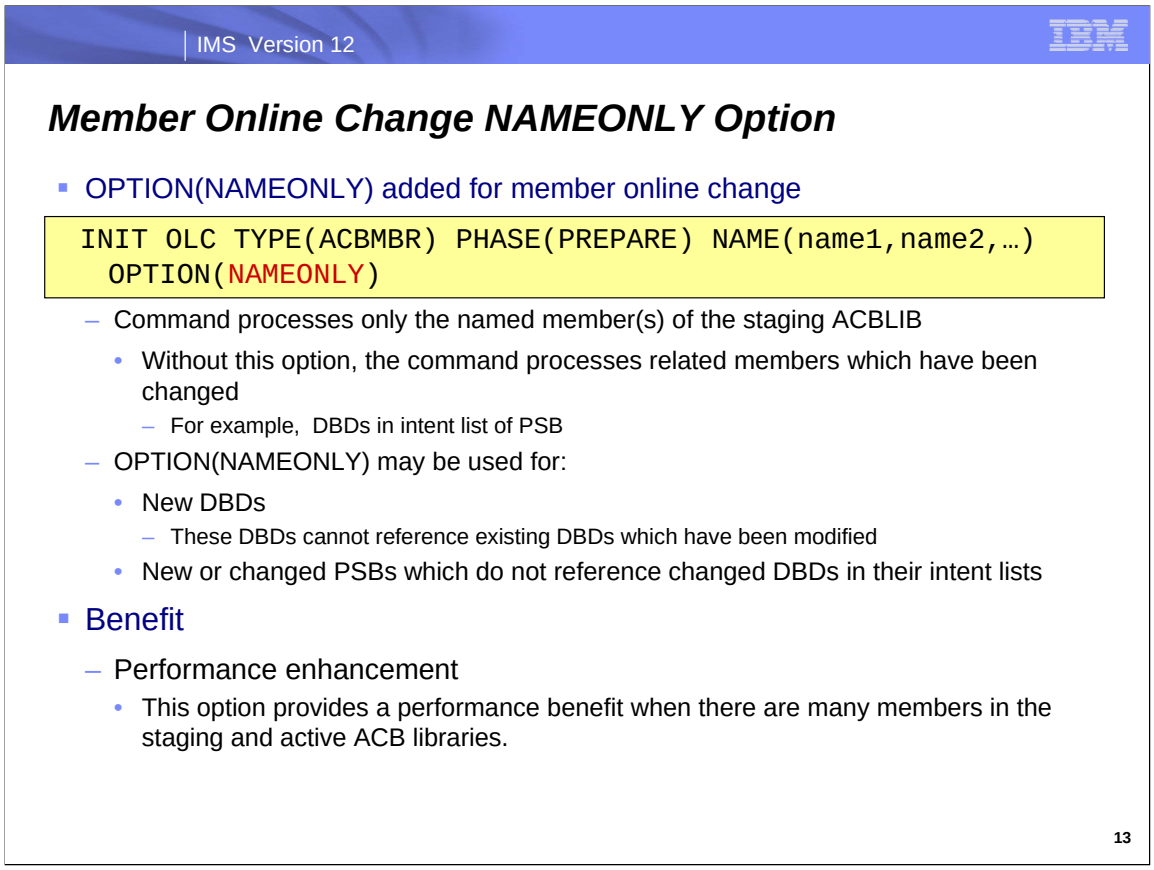
Since software charges are often based on the use of CPs, not zAAP or zIIP processors, the separation of the CPU time into two fields will help users to more accurately reflect the use of processors which affect software charges. This can be reflected in statistics used for accounting and charge out purposes.

Since Java programs can take advantage of zAAP processors, JMP and JBP regions are likely to be the regions most affected by this enhancement.

zIIP support for some CQS processing

- Request response processing for authorized CQS clients is eligible to run on a zIIP
- Examples
  - When the IMS control region is running with IMS Shared Message Queues or Shared IMS Fast Path Message Queues enabled
  - When the IMS Resource Manager address space is using a resource structure

Request Response Processing for authorized Common Queue Server (CQS) clients in IMS 12 is executed under enclave service request blocks (SRBs). In IMS 12 and subsequent releases, IMS will request z/OS to process such work on an available System z Integrated Information Processor (zIIP). Request response processing is the processing of the return of data from the CQS address space to an authorized CQS client address space in response to a request the client directed to the CQS. Authorized CQS clients are those clients that register to IMS 12 CQS while executing in supervisor state and with a "system" Program Status Word (PSW) key (keys 0 through 7).



IMS 12 APAR PM28802 added a new option for member online change. This is the NAMEONLY option. It is specified by including OPTION(NAMEONLY) in the command.

Member online change updates the active ACBLIB from a staging ACBLIB. With OPTION(NAMEONLY) the command copies only the specified member of the staging ACBLIB to the active ACBLIB. Without OPTION(NAMEONLY), the command also processes members related to the specified member. For example, if a PSB is specified and its intent list includes a DBD which has been modified in the staging library, the command copies the specified PSB, the modified DBD and all PSBs which include the modified DBD in their intent lists.

OPTION(NAMEONLY) may be used for new DBDs. These are DBDs which exist in the staging ACBLIB but do not exist in the active ACBLIB. These new DBDs cannot reference existing DBDs which have been modified. A DBD reference another DBD which one is used as a secondary or primary index for the other or when one is logically related with the other.

OPTION(NAMEONLY) may not be used for existing DBDs in the active ACBLIB.

OPTION(NAMEONLY) may be used for new and existing PSBs. These PSBs must not reference modified DBDs in their intent lists.

The use of OPTION(NAMEONLY) provides a performance benefit when there are many members in the staging and active ACB libraries.

Extended Address Volume (EAV) for Non-VSAM Data Sets

- IMS 12 adds EAV support for some non-VSAM system data sets
  - IMS OSAM database data sets
  - Restart Data Set (RDS)
  - Message Queue data sets (queue blocks, long message and short message)
  - Logs (OLDS, SLDS and WADS)
  - IMS SPOOL data sets (UNITYPE=DISK and UNITYPE=SPOOL)
  - BPE External Trace Data Sets
- z/OS 1.12 is required for these data sets
- EATTR=OPT must be specified on allocation JCL
- Large sequential data set support added for IMS SPOOL data sets
  - UNITYPE=SPOOL or UNITYPE=DISK on LINEGRP macro
- Benefits
  - Support the placement of more data sets on a single volume
  - Allow users to manage fewer numbers of larger volumes
  - Less need for multi-volume OSAM

14

Review: Large sequential data set support increases the amount of data for a data set that may reside on a volume. It allows the data set to occupy more than 65,535 tracks on a volume. It does not affect the maximum total size of the data set. It only affects the maximum on a volume. EAV support allows data to be placed at cylinder addresses greater than 65,019. It does not affect the amount of data in a data set. It only affects where it may reside.

IMS 12 adds support for non-VSAM system data sets to reside in the Extended Addressing Space (EAS) of an Extended Address Volume (EAV) volume. This support applies the to data sets listed on the slide. EAVs have more than 65,019 cylinders. The EAS is those cylinders with addressed greater than 65,019 cylinders. This support requires both IMS 12 and z/OS 1.12.

VSAM data sets already had this support with z/OS 1.10 or higher and IMS Version 9 or higher.

The use of EAS requires that the non-VSAM data sets be allocated with EATTR=OPT specified on the JCL. This is not the default for non-VSAM data sets. EATTR=OPT is the default for VSAM data sets.

With this support, users will have the ability to define more data sets on a single volume.

APAR PM30501 for IMS 12 has also added large sequential data set support for IMS Spool data sets. These are IMS online data sets defined with UNITYPE=SPOOL or UNITYPE=DISK on the LINEGRP macro. Large sequential data set support allows more than 65,535 tracks to be allocated on a volume. Without this support, data sets larger than 65,535 tracks required multiple volumes. Large sequential data set support requires that DSNTYPE=LARGE be specified on the JCL when the data set is allocated.

IMS 11 provided support for IMS VSAM data sets to use EAV volumes.

Logging Enhancements

Logging Enhancements

- Extended Format Support for OLDS and SLDS
  - Optional
  - Allows OLDS and SLDS to be striped
    - Increased logging speed
- Log buffers moved above the 2-gigabyte boundary (“bar”) in virtual
  - Optional
  - Frees substantial amount of ECSA
- Simplified WADS management
  - Improved WADS performance

The IMS logger has been enhanced in several ways.

The OLDS and SLDS may be extended format data sets in IMS 12. Extended format allows the use of striping. This can increase the maximum data rates for logging.

Log buffers are optionally moved above the 2 gigabyte boundary in virtual storage. Previous versions of IMS allowed users to place the log buffers above the 2 gigabyte boundary in real storage, but not in virtual storage.

WADS channel program enhancements provide improved performance.

Striping of OLDS and SLDS

- Use of extended format data sets allows striping
  - Striping allows multiple concurrent I/Os for sequential processing
    - Data set is spread across multiple volumes
    - Increased logging rates
- IMS 12 allows OLDS and SLDS to be defined as extended format data sets
  - Extended format is specified with data set type of 'EXT'
    - JCL allocation requires DATACLAS and STORCLAS parameters on DD statement
  - Striping is invoked for extended format data sets when the storage class has Sustained Data Rate (SDR) value of 5 or higher

IMS 12 supports the use of extended format data sets for OLDS and SLDS. Extended format data sets allow striping to be used. With striping the data for a data set is written in multiple "stripes" to different volumes. This provides for greater data rates. When used with IMS logs striping provides increased logging rates.

Extended format data sets require the use of system managed storage and the specification of a data set type of 'EXT' in the data class. Striping is used when a sufficiently high Sustained Data Rate (SDR) is defined in the storage class used for the data set.

To define a storage class for data striping, the storage class must have the Sustained Data Rate attribute set to 5MB/sec or higher when using 3390 devices. SMS determines the number of volumes to use for a striped data set based on the value specified for the Sustained Data Rate attribute in the storage class. The value specified for the SDR is divided by 4 and rounded up to obtain the number of stripes for 3390 devices.

OLDS Buffers Specifications

- OLDS buffers above 2-gigabyte boundary
  - Specified with BUFSTOR=64 on OLDSDEF statement in DFSVSMxx
    - Puts buffers in 64-bit virtual storage
      - Requires that block size is 4K multiple
    - OLDS must be extended format
    - Frees substantial amount of ECSA
- OLDS block size
  - Optionally specified with BLKSZ parameter on OLDSDEF statement in DFSVSMxx member
    - Previously, it was set only from the data set characteristics
  - BLKSZ= is recommended
- If BLKSZ and BUFSTOR=64 are both specified
  - IMS will round up number of buffers (BUFNO=) to a megabyte boundary
    - 64-bit storage is acquired in megabyte increments
    - Maximizes number of buffers in acquired storage

18

The OLDSDEF statement in DFSVSMxx contains parameters to define the OLDS usage. IMS 12 adds the BUFSTOR parameter. BUFSTOR=64 causes IMS to build the OLDS buffers in 64-bit virtual storage. BUFSTOR=31 causes IMS to build OLDS buffers in 31-bit storage. BUFSTOR=31 is the default. IMS 12 also adds the BLKSZ parameter. It is optional. When it is specified, it determines the OLDS block size. Without it, the block size is determined from the data set characteristics. The use of BLKSZ is recommended since it avoids potential problems caused by different data set characteristics for different OLDS.

64-bit storage (above the 2 gigabyte boundary) is allocated in megabyte increments. When BUFSTOR=64 and BLKSZ= are both specified, IMS rounds up the BUFNO value to the a megabyte boundary. This maximizes the number of buffers that are created in the acquired storage.

Migration to Buffers above the 2G Bar

- All OLDS must be extended format for the buffers to be above the 2G bar
- If Logger Exit (DFSFLGX0) or RSR Log Filter Exit (DFSFTFX0) is used
  - Ensure they are capable of handling buffers above 2G
    - The IMS supplied DFSFTFX0 exit is capable
- BUFSTOR=64 must be specified on the OLDSDEF statement
- OLDS block size is multiple of 4K
- IMS must be restarted
  - Restart may be cold or warm

Migration to Striping and Buffers in 64-bit Storage

- First possible set of migration steps to implement striping with OLDS and buffers in 64-bit storage
  1. Define new OLDS data sets with extended formatting
     - Use data class in which data set type is 'EXT'
     - Use storage class with SDR >= 5MB/second
     - Define MDA members for dynamic allocation
  2. Start new OLDS data sets with /START OLDS commands
     - Striping will be used for these data sets
     - A mixture of extended format, basic format and large format OLDS is OK when buffers are below the 2G bar
  3. Stop old OLDS data sets with /STOP OLDS commands
  4. Specify BUFSTOR=64 on OLDSDEF statement
     - Also specify BLKSZ= on OLDSDEF statement
  5. Terminate and restart IMS to get buffers above the bar

This is a possible migration to implement both striping and OLDS buffers in 64-bit storage. Another possible set of migration steps is shown on the next page.

Step 1 defines new OLDS with an SDR sufficient for striping.

Step 2 includes the new OLDS in the online system. Striping will be used when logging is to these OLDS.

Step 3 removes the old OLDS which did not support striping.

Step 4 defines OLDS buffers in 64-bit storage, but this is not yet implemented. The OLDSDEF statement is only read when IMS is started. BLKSZ= is also specified on the OLDSDEF statement. The buffer size will be determined by this parameter.

Step 5 implements buffers in 64-bit virtual storage.

Migration to Striping and Buffers in 64-bit Storage

- Another set of possible migration steps to implement striping with OLDS and buffers in 64-bit storage
  1. Terminate IMS
  2. Rename the OLDS data sets
  3. Define new OLDS data sets using the old data set names
     - Use data class in which data set type is 'EXT'
     - Use storage class with SDR >= 5MB/second
  4. Copy the old OLDS data sets to the new OLDS data sets
  5. Specify BUFSTOR=64 and BLKSZ= on OLDSDEF statement
  6. Restart IMS
     - Striping will be used on all OLDS
     - Buffers will be above the bar

21

This is a possible migration to implement both striping and OLDS buffers in 64-bit storage.

Step 1 terminates IMS.

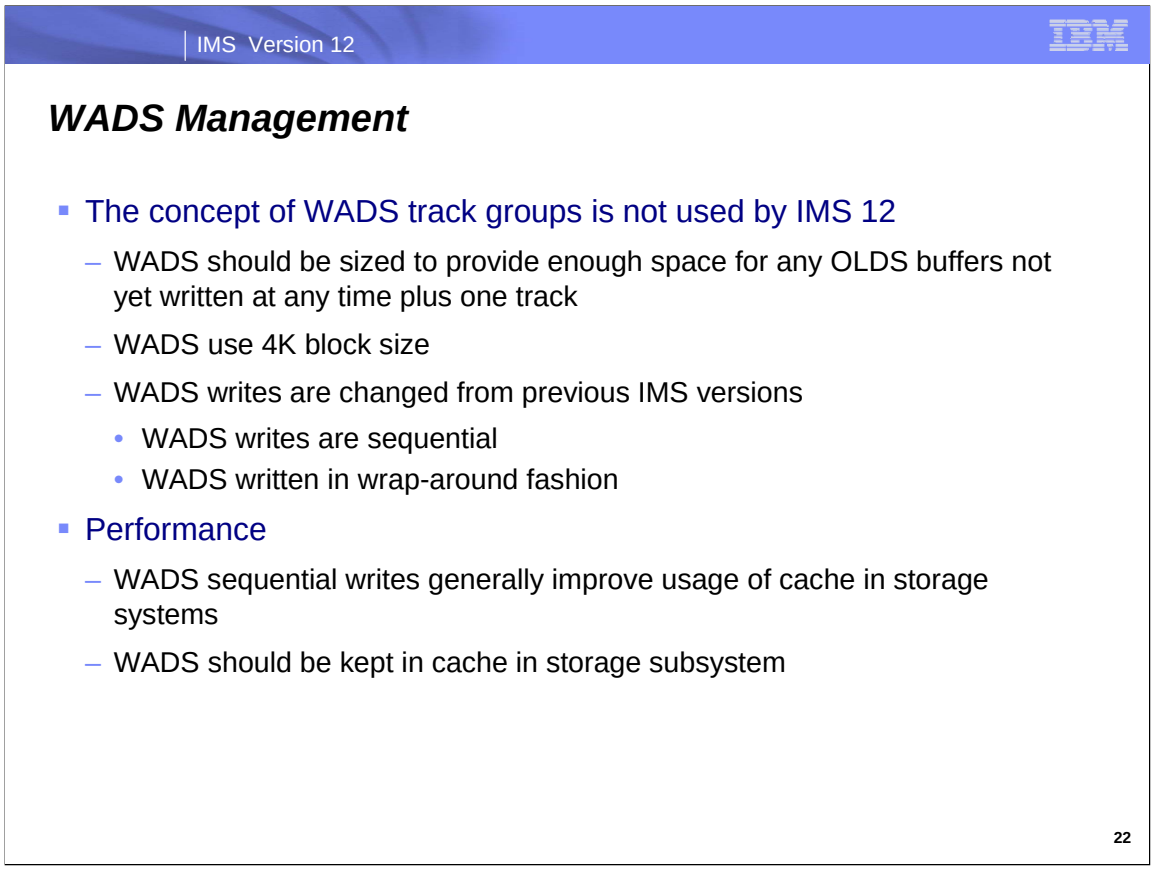
Step 2 defines new extended format OLDS data sets. The newly defined OLDS have an SDR sufficient for striping.

Step 3 renames the OLDS data sets. This is done so that the new OLDS data sets that are defined in step 4 may use the old data set names.

Step 4 copies the old non-extended format OLDS data sets to the new extended format data sets.

Step 5 specifies by use of buffers above the bar in virtual storage. It also specifies the OLDS block size. The buffer size will be determined by the BLKSZ= parameter.

Step 6 warm starts IMS. Since all of the OLDS are now extended format and the OLDSDEF statement includes BUFSTOR=64, the log buffers will be above the 2G bar. Striping will be used for all OLDS.



IMS 12 changes the way that WADS writes are done. The concept of track groups is not used with IMS 12. This changes the calculation for the space required for the WADS and changes the data written by log ahead requests.

In IMS 12 the WADS should be sized to provide enough space for the data in the OLDS buffers which have not yet been written to disk at any time plus one track. This may dramatically reduce the space requirement for the WADS. In previous versions the WADS was sized by using the WADS track group concept. A track group was the OLDS block size/WADS segment size plus 1. A WADS segment size was 2K for OLDS buffers below the bar in real storage and 4K for OLDS buffers above the bar in real storage. The maximum WADS size was (OLDS block size/WADS segment size) + 1) x (number of OLDS buffers) tracks. For example, an installation with 200 24K OLDS buffers above the bar could use a WADS with 1400 tracks. With IMS 12, a system with 200 24K buffers would require no more than enough tracks to hold 200 x 24K plus one track. That would be approximately 101 tracks.

In previous versions of IMS the WADS was written in segments from the OLDS buffers. Successive writes were to different tracks. The scheme is much simpler in IMS 12. Each WADS write is to the next block in the data set. This will generally provide improved performance with modern storage systems. Their algorithms for cache usage generally favor sequential writes. OLDS buffers are conceptually divided into 4K segments. When a write-ahead request is made, the WADS writes include the current 4K segment of the OLDS buffer and any preceding segments which were not previously written. This may include segments in the same OLDS buffer or segments in previous OLDS buffers which have not yet been written to an OLDS data set.

In order to provide the best response times for WADS writes the WADS data should be kept in the cache of the storage subsystem. Since the WADS is written in a wrap-around fashion, this means that the entire WADS should be in the cache for optimum performance.

Logging Enhancements Summary

- Benefits
  - OLDS buffers in 64-bit virtual storage
  - Simplified definition of OLDS block size
  - Increased maximum logging rate
  - ECSA constraint relief

System Pools in 64-bit Real Storage

- The following pools are moved to 64-bit real storage
  - These pools remain in 31-bit virtual storage
  - DBWP – Database work pool
  - DLDP – DMB pool
  - DLMP – CSA PSB pool
  - DPSB – DLI PSB pool
  - PSBW – PSB work pool
- Benefits
  - Reduction in 31-bit fixed real frames for fixed pools
  - Some users will now be able to fix these pools
    - Previously, they were constrained by 31-bit real storage

The pools listed here are moved from 31-bit real storage to 64-bit real storage. They remain in 31-bit virtual storage. The movement to 64-bit real storage will allow some users to fix these pools when they previously could not fix the pools due a shortage of 31-bit real storage. The most likely pools to be affected by this are the CSA PSB pool and the DLI PSB pool since they tend to be the largest pools at most installations.

Sysplex Serial Program Management (SSPM) Disabling

- Optional capability to disable Sysplex Serial Program Management
  - Before IMS 12
    - Serial programs (SCHDTYPE=SERIAL on APPLCTN macro or DEFINE PGM command) enforce serialization of PSB scheduling when:
      - RM is used
      - RM structure is defined
    - Serialization is enforced across the shared queues group
      - Does not apply to CICS or ODBA application schedules
  - IMS 12 adds option not to enforce serialization
    - Option is specified with control statement
      - GBL_SERIAL_PGM=N | Y
        - Specified in DFSDFxxx member COMMON_SERIVCE_LAYER section

        or
        - DFSCGxxx member
      - GBL_SERIAL_PGM=N turns off serialization across the shared queues group

25

IMS 12 adds an option to disable the enforcement of Sysplex Serial Program Management (SSPM) across a shared queues group.

SSPM is normally enforced when the Resource Manager (RM) is used with an RM structure. SSPM forces the serialization of the scheduling of a program across all members of the shared queues group for any program defined with SCHDTYPE=SERIAL on the APPLCTN macro or with SCHDTYPE(SERIAL) on the CREATE PGM command. This serialization applies to the scheduling of MPRs, JMPs, IFPs, BMPs and JBPs. It does not apply to scheduling by CICS or ODBA applications.

This enforcement may be disabled with IMS 12. The disablement is done by including a control statement with GBL_SERIAL_PGM=N in the COMMON_SERVICE_LAYER section of the DFSDFxxx member on in the DFSCGxxx member.

Log Archiving for Non-Recoverable Databases

SPEs: IMS 10 PM18093; IMS 11 PM19363; IMS 12 PM54945

- Archive (DFSUARC0) is enhanced to write “undo” records for non-recoverable full function databases
  - Previously, “undo” records were not archived
  - New control statement option not to archive “undo” records
    - Required to write these records as was done previously
- Non-recoverable database logging:
  - “After image” log records are not written for non-recoverable databases
  - “Undo” log records are written to OLDS for non-recoverable full function databases
    - This allows uncommitted updates to be backed out
  - “Undo” log records are not archived from OLDS before this enhancement
    - They are archived with this enhancement unless overridden by control statement

26

An SPE for IMS 10 (PM18093), IMS 11 (PM19363), and IMS 12 (PM54945) changes archiving for non-recoverable databases. By default, these SPEs cause archive to write the “”undo” or “before image” type x’50’ log records for full function databases. Previously, these records were not written by archive.

A control record allows users not to archive the “undo” records. This mimics the actions of previous IMS versions.

When a database is specified as non-recoverable in DBRC, “after image” log records are not written for it. “Undo” log records are written for full function database so that uncommitted updates may be backed out when a application abends or when IMS fails and is emergency restarted. Before this enhancement, “undo” log records were not archived from the OLDS to SLDS.

Log Archiving for Non-Recoverable Databases

- Problem addressed by this enhancement
  - Online backout using archived log stops when it encounters a “missing” log record
    - Backout is incomplete, other database updates may not be backed out and will be stopped
  - Batch backout using archived log does not backout updates to non-recoverable database but sets no flags in DBRC to prevent its use
- Control statement not to archive non-recoverable database log records
  - `SLDS CMPRSNR((dbd1,ddn1)(dbd2,ddn2),...|ALL)`
- Benefit
  - Eliminates potential database integrity and operational problems

27

The change in archiving was done to avoid a data integrity exposure. If a back out is done and some of the log records it needs were on an OLDS which has been reused since the records were written, the archived SLDS must be used as input to the back out. Back out by online systems reads the “undo” log records. These records are chained. When the “undo” log records for a non-recoverable database are archived, they are replaced with a dummy record (type x’43’). This breaks the chain. When back out encounters one of these dummy records, it must terminate. This means that the back out of all databases, not just the non-recoverable database, may be incomplete. IMS stops these databases. Archiving the “undo” records instead of replacing them with dummy records closes this integrity and operational exposure.

Another potential problem exists with batch back out. Batch back out reads the log forward to collect the database change log records for back out. It backs out the recoverable databases. Since records for the non-recoverable database are not on the SLDS (without this SPE), it does not back out changes for it. This is understandable, but batch back out does not set any flags in DBRC to prevent the use of the database which has not been backed out correctly. This could be a problem.

By default, IMS will archive the before image log records for non-recoverable databases when the SPE is applied. This solves both problems. The SLDS control statement for the Archive utility (DFSUARC0) may be used to specify that the before images are not archived. This is done with the CMPRSNR (compress non-recoverable) parameter. Database data sets are specified by database name and DDNAME. If you want don’t want this archive for any non-recoverable databases, you may specify ALL. Specifying ALL is compatible with the may archive worked before this enhancement. Of course, if these records are not archived, the user is responsible for database integrity.

Command Enhancements

- Enhancements to existing commands and new commands
  - CQS trace command enhancements
  - DBRC command enhancements
  - Dynamic database buffer pool command enhancements
  - Dynamic resource definition (DRD) command enhancements
  - Fast Path secondary index command enhancements
  - HALDB command enhancements
  - IMS Connect command enhancements
  - MSC command enhancements
  - IMS repository function command enhancements
  - OTMA command enhancements

IMS Version 12 includes enhancements to existing commands, as well as new commands. Most of the enhancements are provided in type-2 commands, to support the IMS strategy of enhancing the capability of single point of control (SPOC) applications that issue type-2 commands through the Operations Manager (OM) API or the REXX SPOC API.

Details of these command enhancements are in Chapter 1 of IMS 12 Resource Planning.

Command Enhancements

- Enhancements are focused on type-2 commands for the Operations Manager (OM) environment

- Benefits
  - Support of new IMS 12 functions
  - Improved manageability

System Usability Enhancements

- IMS 11 SPE User Exit Enhancements
  - Enables support for multiple instances of a user exit type
  - Provides support for 3 new exits
  - Implements the QUERY USEREXIT command
  - Included in IMS V12

IMS 11 SPE User Exit Enhancements

IMS 11 APAR PM04456 / PTF UK67199

- Ability to run multiple instances of a supported user exit type
  - No longer need user or vendor written code to manage multiple exits
- Three new IMS user exit types
  - IMS Initialization/Termination **- INITTERM**
  - IMS CQS Event **- ICQSEVNT**
  - IMS CQS Structure Event **- ICQSSTEV**
- Existing User Exits eligible to use new exit services
  - Product Partner Exit DFSPPUE0 - **PPUE**
  - Restart Exit DFSRSTX0 (available in IMS 10) – **RESTART**
- Implements the QUERY USEREXIT command

31

This IMS V11 maintenance enables IMS to support multiple instances of a user exit type without the need for customer or vendor written code. An IMS customer can run multiple tools that use the same exit point without the need for additional software to manage the multiple exits. Only certain supported exit types are managed by this maintenance. These functions are included in IMS V12.

New user exit types were added with this IMS 11 maintenance:

The Initialization/Termination exit type provides access to IMS during early IMS initialization and during normal / abnormal IMS termination.

The IMS CQS Event exit type is called when IMS receives notification of a CQS event such as CQS has terminated.

The IMS CQS Structure Event exit type is called when IMS receives notification of a structure related event from CQS such as a structure rebuild.

The EXITDEF parm in the USER_EXITS section of DFSDFxxx must be used to define the exit type and the exit routine(s). The routines are called in the order that they are listed in the EXITDEF parameter.

Support for new user exit services is added for the Product Partner Exit (DFSPPUE0) and the Restart Exit (DFSRSTX0). The customer has the choice to continue to use the old user exit services or to use the new user exit services which give the capability to define multiple user exit routines that are called at the exit points Other existing user exit types continue to be supported as they are today.

There is a new QUERY USEREXIT command to view exit details.

IMS 11 SPE User Exit Enhancements

- IMS Initialization/Termination Exit
  - Available to all IMS regions: TM/DB, DBCTL, DCCTL, and FDBR
  - Exit called during …
    - Early IMS initialization
    - Normal and abnormal IMS termination
  - Exit routine(s) must be defined in the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member
    - EXITDEF=(TYPE=**INITTERM**,EXITS=(***exitnames***))
    - Exit routines called in the order listed
    - No default exit name
  - Exit load module must be included in an authorized library in the JOBLIB, STEPLIB, or LINKLIST concatenation
  - This exit is eligible to use IMS Callable Services

32

The IMS Initialization/Termination exit is called during early IMS initialization, during normal / abnormal IMS termination, before the RESTART exit. The exit is available to all IMS regions including DB/DC, DBCTL, DCCTL, and FDBR.

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

Multiple exit routines can be defined to IMS using the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member. The routines are called in the order that they are listed in the EXITDEF parameter

This exit routine uses the current version of the standard exit parameter list. See Chapter 1 of the IMS V11 Exit Routine Reference for the current version and the contents of the standard exit parameter list.

The Initialization/Termination exit routine parameter list is mapped by DFSITXP.

IMS 11 SPE User Exit Enhancements

- IMS CQS Event Exit
  - Available to IMS regions: TM/DB and DCCTL
  - Exit called when IMS receives notification of a CQS event
  - Exit routine(s) must be defined in the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member
    - EXITDEF=(TYPE=**ICQSEVNT**,EXITS=(***exitnames***))
    - Exit routines called in the order listed
    - No default exit name
  - Exit load module must be included in an authorized library in the JOBLIB, STEPLIB, or LINKLIST concatenation
  - This exit is eligible to use IMS Callable Services

33

The IMS CQS Event Exit is called when IMS receives notification of a CQS event, such as CQS termination or a taskconnects to CQS. IMS is notified of the events from CQS via an IMS exit routine that is driven in the IMS address space by CQS. The IMS CQS Event Exit exit is only available to an IMS region that registers with CQS which can include DB/DC and DCCTL.

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

Multiple exit routines can be defined to IMS using the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member. The routines are called in the order that they are listed in the EXITDEF parameter

This exit routine uses the current version of the standard exit parameter list. See Chapter 1 of the IMS V11 Exit Routine Reference for the current version and the contents of the standard exit parameter list.

The IMS CQS Event exit routine parameter list is mapped by DFSCEXP.

IMS 11 SPE User Exit Enhancements

- IMS CQS Structure Event Exit
  - Available to IMS regions: TM/DB and DCCTL
  - Exit called when IMS receives notification of a CQS structure event
  - Exit routine(s) must be defined in the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member
    - EXITDEF=(TYPE=**ICQSSTEV**,EXITS=(***exitnames***))
    - Exit routines called in the order listed
    - No default exit name
  - Exit load module must be included in an authorized library in the JOBLIB, STEPLIB, or LINKLIST concatenation
  - This exit is eligible to use IMS Callable Services

The IMS CQS Structure Event Exit routine is called when IMS receives notification of a CQS structure event, such as a structure rebuild or structure overflow. IMS is notified of the structure events from CQS via IMS's CQS structure exit that is driven in the IMS address space by CQS. The exit is only available to an IMS region that registers with CQS which can include DB/DC and DCCTL.

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

Multiple exit routines can be defined to IMS using the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member. The routines are called in the order that they are listed in the EXITDEF parameter.

This exit routine uses the current version of the standard exit parameter list. See Chapter 1 of the IMS V11 Exit Routine Reference for the current version and the contents of the standard exit parameter list.

The IMS CQS Structure Event exit routine parameter list is mapped by DFSCSXP.

IMS 11 SPE User Exit Enhancements

- Partner Product Exit – DFSPPUE0
  - Support added for the new user exit services
    - Capability to define multiple user exit routines called at the PPUE exit point
    - Capability to use QUERY USEREXIT for PPUE
  - Available to IMS regions: TM/DB, DBCTL and DCCTL
  - Exit routine(s) must be defined in the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member
    - EXITDEF=(TYPE=**PPUE**,EXITS=(***exitnames***))
    - Exit routines called in the order listed
    - No default exit name
  - Exit can be named DFSPPUE0 and must be linked into a library in the STEPLIB concatenation
  - Can continue using old user exit services for PPUE

35

Support for the new user exit services is added for the Product Partner Exit (DFSPPUE0). The user has the choice to continue to use the old user exit services or to use the new user exit services which give the capability to define multiple user exit routines that are called at the PPUE exit point.

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

Multiple exit routines can be defined to IMS using the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member. The routines are called in the order that they are listed in the EXITDEF parameter.

The Partner Product Exit can continue to use the old user exit services, but you will not be able to load multiple exit routines and PPUE will not be supported with the QUERY USEREXIT.

IMS 11 SPE User Exit Enhancements

- Restart Exit
  - Support added for the new user exit services
    - Capability to define multiple user exit routines called at RESTART exit point
    - Capability to use QUERY USEREXIT for RESTART
  - Available to IMS regions: TM/DB, DBCTL, and DCCTL
  - Exit routine(s) must be defined in the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member
    - EXITDEF=(TYPE=**RESTART**,EXITS=(***exitnames***))
    - Exit routines called in the order listed
    - No default exit name
  - Exit load module must be included in an authorized library in the JOBLIB, STEPLIB, or LINKLIST concatenation
  - Can continue using old user exit services for RESTART

The Restart exit, which was available in IMS 10, can take advantage of the new exit services.

Support for the new user exit services is added for the Restart Exit (DFSPPUE0). The user has the choice to continue to use the old user exit services or to use the new user exit services which give the capability to define multiple user exit routines that are called at the RESTART exit point.

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

Multiple exit routines can be defined to IMS using the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member. The routines are called in the order that they are listed in the EXITDEF parameter.

The Restart Exit can continue to use the old user exit services, but you will not be able to load multiple exit routines and RESTART will not be supported with the QUERY USEREXIT.

IMS 11 SPE User Exit Enhancements

- QUERY USEREXIT Command
  - Type-2 command to display information about user exits defined in the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member
  - Command is not available for user exits not defined in EXITDEF parameter
  - Default command routing to all IMSs in the IMSplex
  - Sample information returned for requested user exit type:

    Module Name(s) Entry Point Address
    Module Size Number of Calls (since initialization)
    Total Elapsed Time Spent in Exit Routine (since initialization)
    IMS member name that created the output for the exit
  - Security authorization
    - Resource name is ***IMS.plxname.QRY.USEREXIT***
    - Consider setting similar to other OM QUERY commands
  - Examples in Appendix for User Exit Enhancements

37

The QUERY USEREXIT command is a type-2 command that allows the user to display information about the user exits that are defined in the USER_EXITS section of the DFSDFxxx member. The output contains an entry for each user exit module within each user exit type named in the QUERY.

User exits that are not defined in the DFSDFxxx member are not included in the output of the command.

The QRY USEREXIT command is routed to all IMSs in the IMSplex as its default routing.

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

The QUERY USEREXIT is supported in TM/DB, DBCTL and DCCTL environments.

The command output for QUERY USEREXIT is defined in XML and is available to automation programs which communicate with OM.

Consider securing the QUERY USEREXIT command similar to other QUERY commands.

IMS 11 SPE User Exit Enhancements

- Miscellaneous Information
  - Sample exit code provided in IMS.SDFSSMPL

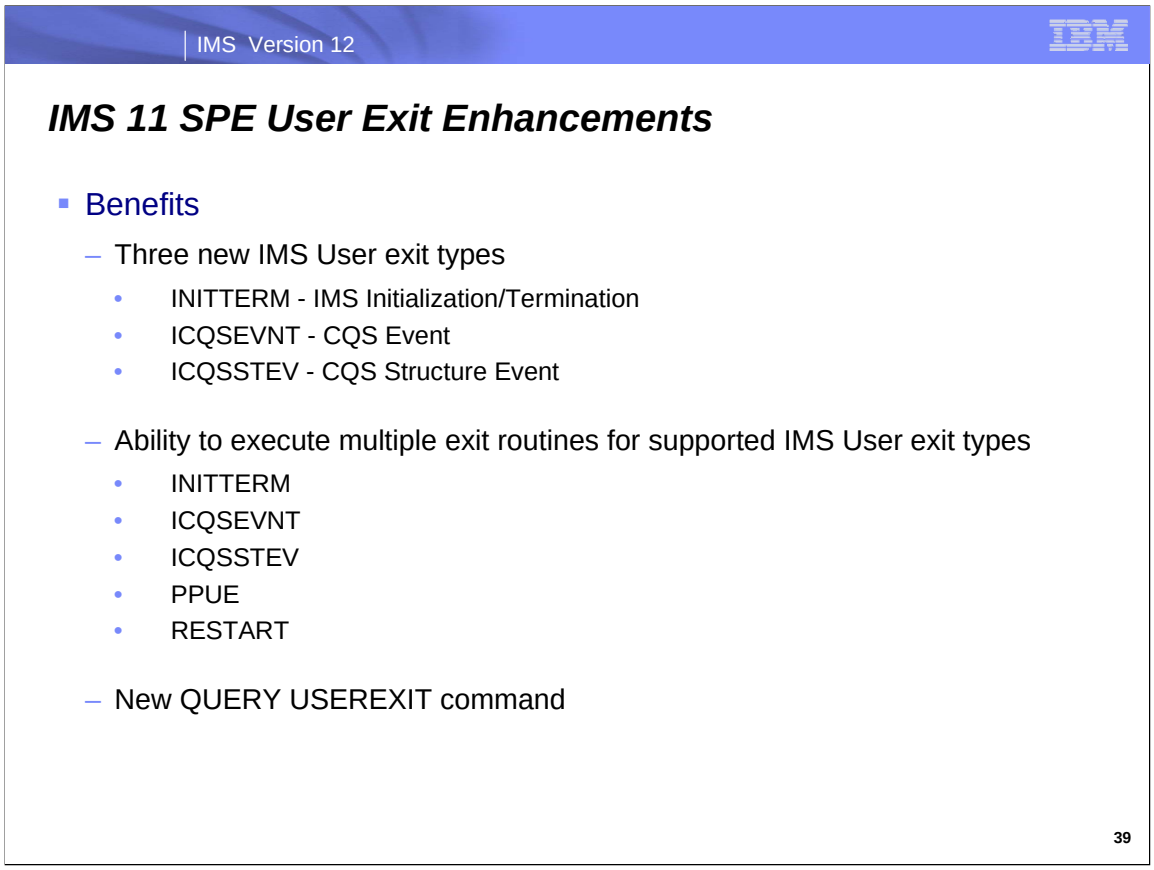
Exit Type	Sample Code
INITTERM	DFSITMX0
ICQSEVNT	DFSCQEX0
ICQSSTEV	DFSCSEX0
PPUE	DFSPPEX0
RESTART	DFSRSTX0

  - New x'4517' log record
    - Log record created at checkpoint
    - Mapped by DSECT DFSL4517
    - Contains relevant user exit data
  - Further information at Technote: https://www.ibm.com/support/docview.wss?uid=swg27021341

38

Sample exit code provided in:
IMS.SDFSSMPL(DFSITMX0)
IMS.SDFSSMPL(DFSCQEX0)
IMS.SDFSSMPL(DFSCSEX0)
IMS.SDFSSMPL(DFSPPEX0)
IMS.SDFSSMPL(DFSRSTX0)

New x'4517' LOGREC will be cut at checkpoint time and contain user exit data. It is mapped by DSECT DFSL4517.



New exit points within IMS Control Region to drive a user exit during IMS initialization and termination, CQS events and CQS structure events.

Several IMS exits now defined by exit type and now support the calling of multiple exit routines for an exit point:

INITTERM
ICQSEVNT
ICQSSTEV
PPUE
RESTART

The QUERY USEREXIT command is provided so users can display the status of exit points.

RAS Enhancements

- IMS 12 CQS Traceability
  - Creates two new BPE trace tables to track CQS structure events
- IMS Dump Formatter Enhancements
  - Support for the Repository Server and the Repository Client address spaces
  - Support for the OTMA C/I
- Reduce IMS Module Aliases in RESLIB
  - Begins a process to reduce IMS load module aliases
- End-Of-Memory / End-Of-Task (EOT) Tracing Facility
  - Created a step trace function in the End-of-Task (EOT) process

There are several enhancements in the IMS RAS area.

There are two new BPE tace tables for CQS.

The IMS Dump Formatter Enhancements include support for the repository server address space to the Other IMS Components (6) section and the repository client address space to the Other IMS-Related Products (7) section. Support has been added for OTMA C/I to the Other IMS Components (6) section.

With IMS 12, we begin a process to reduce IMS module aliases in RESLIB where possible.

An End-of-Task (EOT) step trace facility, similar to that employed by the End-of-Memory (EOM) process, has been created to trace the step-by-step flow through the EOT process. This is accomplished by updating a nibble trace double-word in the region's IDT (IDTEOTTR) with a unique marker for each successful step processed and/or for each decision point reached and evaluated. Message DFS0798I should be issued at the end of the EOT process, if at all possible, to externally document the results of the process.

RAS Enhancements

- IMS 11 SPE BPE Trace Selective Initialization
  - Creates a new BPE trace table feature that allows a trace table to be defined to require explicit commands
    - IMS Connect Recorder Trace, RCTR, trace table definition has been updated to enable the new explicit commands feature
- New /DIAGNOSE Command SET AOSLOG Function
  - Enable and disable log record capture of events related to APPC and OTMA synchronous transactions in a shared-queues environment

There is a new selective initialization feature for the BPE trace tables. This applies to the IMS Connect BPE Recorder Trace.

The /DIAGNOSE Command Enhancements address the need to improve diagnostic information and to streamline the problem determination process. This line item component will enhance the /DIAGNOSE command SET AOSLOG function. It will allow for dynamic enabling and disabling of the capture of events related to APPC and OTMA synchronous transactions in a shared queues environment. If enabled, x'6701' log records will be written from the FE system when the BE system's message is received.

RAS Enhancements

- /DIAGNOSE Command SNAP Function Enhancements
  - Added six new resource types to the SNAP function
  - Provided DISPLAY option to route output back to issuing LTERM
  - Provided LIMIT option to restrict number of lines of output going to LTERM
  - Provided SHOW parameter to control type and amount of output produced

The /DIAGNOSE Command Enhancements address the need to improve the reliability of diagnostic information and to streamline the problem determination process. This line item component will enhance the /DIAGNOSE command SNAP function by: adding six new SNAP resource types – AREA, DB, LINE, LINK, PGM, and REGION – which support multiple resource name parameters; adding a display output option; and providing an output filtering option.

The overall benefits of these enhancements are improved problem diagnosis and resolution.

IMS 12 CQS Traceability

- Background
  - IMS Shared Queues environments can produce a high volume of trace data
    - Can quickly fill in-core trace table
    - Table wrap around technique can cause the loss of important trace data
  - Need to preserve trace entries for structure events
    - Lack of trace data can slow down the diagnostic process
- Solution
  - Create two new BPE trace tables to track CQS structure events
- Benefits
  - Preserve more trace entries for CQS structure events
  - Improve the time to diagnose CQS problems

43

IMS systems with shared queues and CQS can produce a high volume of trace data that quickly fills up the incore trace storage. The wrap around tracing process can cause the loss of important trace data. There is a need to preserve trace data since missing trace data can slow down problem resolution. This enhancement creates 2 new BPE trace tables to preserve structure events trace entries. The size of the trace records has been increased to boost trace data storage capacity.

This will help to provide the IMS support team with valuable diagnostic data for customer problem resolution. Shared queues related problems can possibly be resolved more quickly.

IMS 12 CQS Traceability

- New CQS Trace Tables
  - BPE based tracing
  - Track CQS structure events based on structure event characteristics
    - Structure Event trace table (SEVT) tracks structure events
    - Structure Overflow trace table (OFLW) tracks overflow events
  - The existing CQS structure trace table (STR) will track client request activity only
    - No longer tracks structure events

Two new BPE based trace tables will track structure events separately.

Based on CQS structure event characteristics, the following two new trace tables are created:

- Structure Event trace table (SEVT) to trace all of the structure events except the overflow events
- Structure Overflow trace table (OFLW) to trace the overflow events

The existing CQS Structure trace table (STR) will track the client request activity only and no longer track the structure events.

IMS 11 SPE BPE Trace Selective Initialization

- Background
  - IMS Connect Recorder Trace facility delivered in IMS V11 uses a BPE trace table
  - Inadvertent activation of BPE tracing is possible
    - If BPE trace initialization encounters a generic type (*) on the TRCLEV statement in the HWS BPE configuration member:

      TRCLEV=(*,HIGH,HWS)
    - Performance problems could result from running Recorder Trace inadvertently
  - Different than activating the original IMS Connect Recorder Trace
    - Explicit IMS Connect “OPEN” command

45

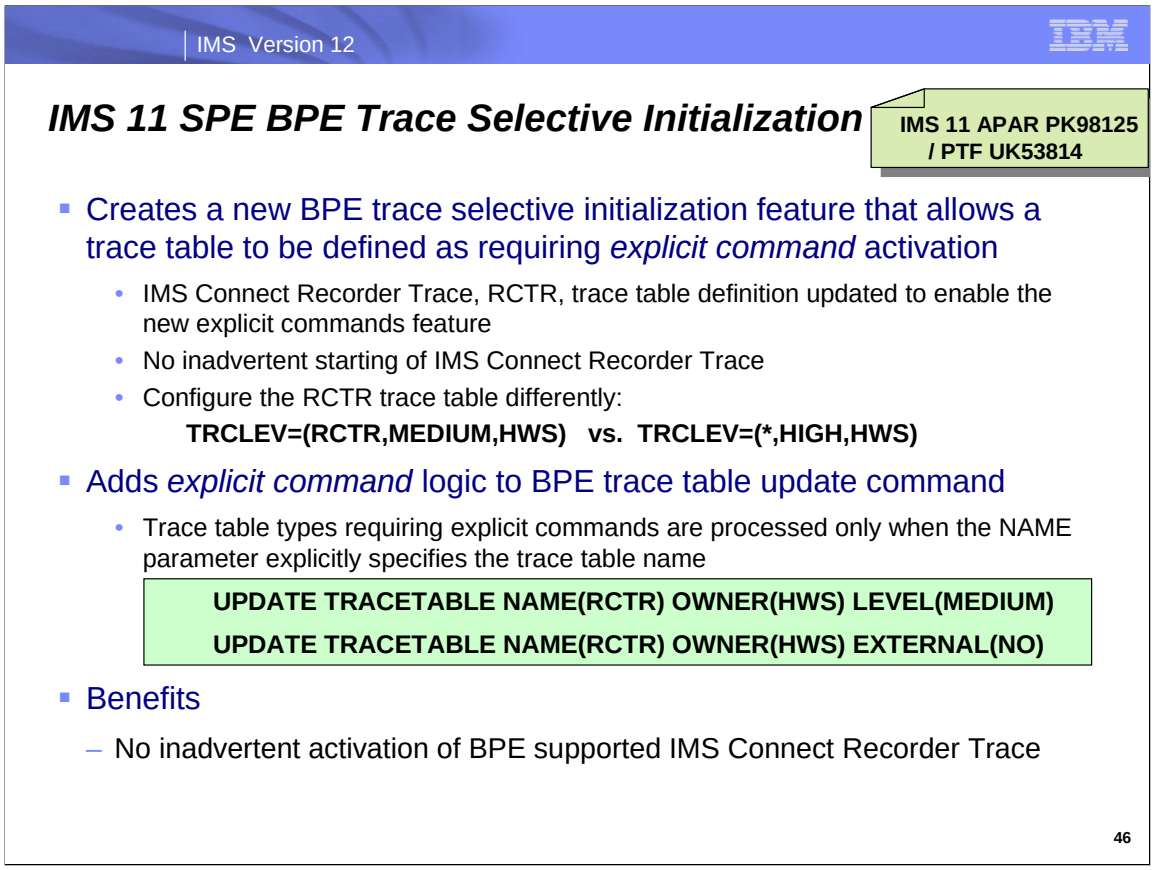
The IMS 11 Recorder Trace facility can be inadvertently activated when BPE trace initialization encounters a * for type on the TRCLEV statement in the HWS BPE configuration member.

The Recorder Trace facility delivered in IMS 11 uses a BPE trace table. If a generic type is coded on the TRCLEV statement in the HWS BPE configuration member:

TRCLEV=(*,HIGH,HWS)

The new Recorder Trace facility will be automatically activated. The unintended overhead of running Recorder Trace inadvertently may result in performance problems.

This is operationally different from the original Recorder Trace facility which required an operator OPEN command and may result in the new Recorder Trace facility being started inadvertently.



The BPE trace selective initialization function delivered with this APAR is meant to address this issue by creating a new BPE trace table feature. This new feature allows a trace table to be defined to require explicit commands. The RCTR trace table definition, in module HWSTTH00, has been updated to enable the new explicit commands feature to avoid starting Recorder Trace inadvertently.

A trace table type which requires explicit commands will not be processed if a * is specified for the TRCLEV type parameter. To configure a trace table type which requires explicit commands a separate TRCLEV statement which explicitly identifies the table must be specified:

TRCLEV=(RCTR,MEDIUM,HWS)

The following TRCLEV statement will start all IMS Connect trace tables with level HIGH with the exception of the RCTR table:

TRCLEV=(*,HIGH,HWS)

To activate the RCTR trace table in addition to the other trace tables use these two TRCLEV statements:

TRCLEV=(*,HIGH,HWS) and TRCLEV=(RCTR,MEDIUM,HWS)

Explicit command logic has also been included in the BPE UPDATE TRTAB command. Trace table types that require explicit commands, like the IMS Connect RCTR trace, will not be processed if a generic name - NAME(*) - or a wildcard pattern - NAME(R%%%) - is specified for the trace table name. Trace table types which require explicit commands are processed only when their name is explicitly specified with the NAME parameter:

UPDATE TRACETABLE NAME(RCTR) OWNER(HWS) LEVEL(NONE)
UPDATE TRACETABLE NAME(RCTR) OWNER(HWS) LEVEL(MEDIUM)
UPDATE TRACETABLE NAME(RCTR) OWNER(HWS) EXTERNAL(NO)
UPDATE TRACETABLE NAME(RCTR) OWNER(HWS) EXTERNAL(YES)

There is no longer a problem with inadvertent activation of the IMS Connect Recorder Trace.

IMS Dump Formatter Enhancements

- Dump Formatter Support for IMS Repository
  - Support added to IMS Dump Formatter for the Repository Server address space
    - Via the Other IMS Components (6) section of the Dump Formatter
  - Support added to IMS Dump Formatter for the Repository Client address space
    - Via the Other IMS-Related Products (7) section of the Dump Formatter
- OTMA C/I Serviceability Enhancements
  - Currently has no dump formatting or tracing capabilities
  - Enhances the IMS Dump Formatter by adding support for OTMA C/I
    - Via the Other IMS Components (6) section of the Dump Formatter

47

Dump Formatter support is added for the new IMS 12 Repository RS address space.

OTMA C/I currently has no dump formatting or tracing capabilities making it difficult to debug problems. This new support will assist the IMS support center in debugging OTMA C/I problems.

Reduce IMS Module Aliases in RESLIB

- Reduced module aliases in RESLIB, where possible
- For remaining aliases, doc provided on module to alias dependencies
- Should help reduce errors when implementing selected service
  - Fewer aliases that can be overlooked when modules manually copied

Various IMS load modules contain aliases, which make them error prone when service affects these modules. Common practice when implementing selected service is to individually copy modules affected by the service. When modules being copied contain aliases it is very easy to overlook this and not propagate the aliases.

EOM / EOT Tracing Facility Enhancement

- Created a step trace function in the End-of-Task (EOT) process similar to that employed by the End-of-Memory (EOM) process
  - Traces the step-by-step flow through the EOT process
    - Updates a nibble trace in the region's IDT with a unique marker
      - For each successful step processed
      - For each decision point reached and evaluated
  - Message DFS0798I is issued at the end of the process to externally document the results of EOT processing

This enhancement will improve problem debugging.

New /DIAGNOSE Command SET AOSLOG Function

- Support for the APPC / OTMA Synchronous Shared Queues enhancement to use XCF communications for IMS systems in a SQ Group
  - AOSLOG=Y|N is a new keyword in the DFSDCxxx proclib member
    - applicable to a Front End system only
  - Enable or disable the logging of x'6701' records to capture events related to APPC / OTMA synchronous transactions in a shared-queues environment
- IMS Log Records
  - Front End system writes x'6701' log trace record when response message from Back End system is received
  - New ID of "TIB3" is added to the x'6701' log record
- Benefits
  - Enhanced problem diagnostic capability and quicker problem resolution

50

A new keyword AOSLOG=Y|N has been added to the DFSDCxxx PROCLIB member to specify whether the FE system will write a x'6701' log record for the following cases:

- Response message returned from the BE system via XCF for transactions with all sync levels of NONE, CONFIRM and SYNCPT.
- Error message returned from the BE system via XCF for transaction with all sync levels of NONE, CONFIRM and SYNCPT.

Y: indicates that the FE system will log the 6701 record.

N: indicates that the FE will not log any 6701 record. This is the default value if omitted.

The AOSLOG parameter is only applicable to the FE system. The BE system is not required to specify this parameter.

The FE system will write a x'6701' trace record when it receives a response message from the BE system via XCF for transactions with all sync levels of NONE, CONFIRM, and SYNCPT if the AOSLOG logging is enabled. A new ID of TIB3 is added to the x'6701' log record.

The enhancement will provide good diagnostic information and enable easier problem resolution.

New /DIAGNOSE Command SNAP Functions

- Six new SNAP resource types

Resource Type	SNAP Captures…
AREA	control block information for fast path areas
DB	control block information for full function databases
LINE	control block information for communication lines
LINK	control block information for MSC logical links
PGM	control block information for programs
REGION / JOBNAME	control block information for dependent regions

  - Support for multiple resource name parameters separated by a comma or a blank

```
/DIAGNOSE SNAP AREA(areaname,areaname,areaname)
/DIAGNOSE SNAP DB(dbname dbname dbname)
```

51

The /DIAGNOSE command SNAP function is used to take a snap shot of system resources at any time without impacting IMS. This minimizes the user's time and effort to provide problem determination data to IBM support. The output can be quickly gathered and transmitted to IBM which avoids the overhead of capturing and transferring a dump. The user's time and effort to provide problem determination data to IBM support should decrease.

There are 6 new SNAP resource types:

- AREA for fast path areas
- DB for full function databases
- LINE for communication line
- LINK for MSC logical links
- PGM for programs
- REGION / JOBNAME for dependent regions

The resource type names requested can be separated by a comma or a space.

RAS Enhancements

- Benefits
  - Diagnose problems interactively
  - Non-disruptive alternatives to producing console dumps
  - Reduce time and effort to capture and analyze IMS diagnostic information
  - Help reduce time to resolve problems
  - Satisfy several customer and IBM requirements
- Appendix for RAS Enhancements has further details

The overall benefits of these enhancements are improved problem diagnosis and resolution.

MIPS reduction enhancements

- CICS threadsafe support
- CICS and ODBA users have new DFSRAS00 exit capability to designate a user as trusted, bypassing RACF or equivalent security checks
- Usage of newer, more efficient hardware instructions when available
  - Long displacement facility / Store Clock Fast (STCKF) facility
- Replacing GETMAIN storage allocation calls with more efficient IMS internal storage management calls for APPC/OTMA scheduling
- Supporting native 64-bit invocation of several highly used IMS internal macro services, reducing AMODE switching for 64-bit modules
- Efficiency/pathlength reductions in CQS inform exit processing and OTMA processing
- Improving IMS shutdown time by reducing OTMA/APPC shutdown quiesce waits
- Benefits – Improved Performance

Improving performance is a major focus for IMS 12.

Summary of System Enhancements …

- Dynamic resource definition (DRD) enhancements
- zAAP/zIIP times in accounting log records
- zIIP support for some CQS processing
- Member online change NAMEONLY option
- EAV support for non-VSAM data sets
- IMS logging enhancements
  - Extended-format data set support for OLDS/SLDS
  - IMS log buffers in 64-bit virtual storage
  - Simplified WADS management
- System pools in 64-bit real storage
- Sysplex Serial Program Management (SSPM) Disabling
- SPE for log archiving of non-recoverable databases

IMS 12 includes several system enhancements that will be discussed in the topic.

Summary of System Enhancements

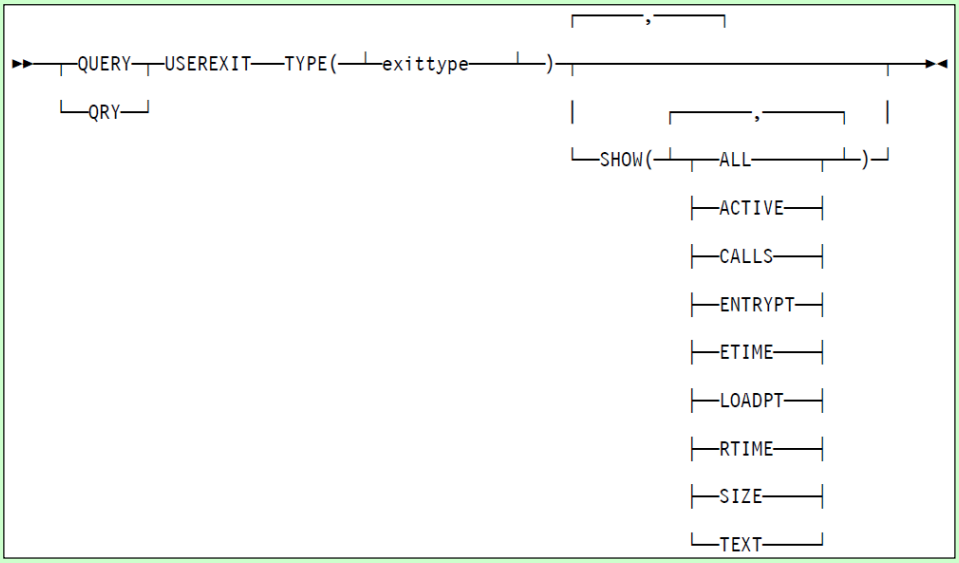
- Command enhancements
- System usability enhancements
  - IMS 11 SPE User Exit Enhancements
- RAS enhancements
  - CQS traceability enhancements
  - IMS Dump Formatter enhancements
  - Reduce IMS Module Aliases in RESLIB
  - End-Of-Memory / End-Of-Task (EOT) Tracing Facility
  - IMS 11 SPE BPE Trace Selective Initialization
  - /DIAGNOSE SNAP command enhancements
- MIPS reduction enhancements

Appendix

User Exits Enhancements

IMS 11 SPE User Exit Enhancements

- QUERY USEREXIT Command Syntax



57

Query command syntax:

QUERY USEREXIT TYPE(PPUE) SHOW(ALL)

The TYPE parameter contains the supported exit type names

The SHOW parameter controls the amount of data returned from the command.

IMS 11 SPE User Exit Enhancements

- QUERY USEREXIT Command

QUERY USEREXIT TYPE(*exittype*)SHOW(*keywords*)

  - TYPE() specifies the user exit type(s) for which information will be displayed
    - Single type or a list of types separated by commas or "*"
      - INITTERM (Initialization/Termination Exit)
      - ICQSEVNT (CQS Event Exit)
      - ICQSSTEV (CQS Structure Event Exit)
      - PPUE (Partner Product Exit)
      - RESTART (Restart Exit)
  - If SHOW() is not specified
    - Only exit type, exit module name, IMS member name and CC for the specified exit type(s) are returned
  - If SHOW() is specified
    - Always returns exit type, exit module name, IMS member name and the CC for the specified user exit type(s)
    - Also returns information specifically requested for the specified user exit type(s)

58

The QUERY USEREXIT is a type-2 command that displays information about user exits. The command can only be specified through the Operations Manager (OM).

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

The valid user exit types are the following:

INITTERM	Initialization/Termination Exit
ICQSEVNT	CQS Event Exit
ICQSSTEV	CQS Structure Event Exit
PPUE	Partner Product Exit
RESTART	Restart Exit

If the SHOW keyword is not specified, only the exit routine name, IMS member name and CC for the specified types are returned.

If the SHOW keyword is specified, specific information requested about the user exit routines will be returned in the output of the command response. The exit type, the exit routine name, IMS member name and CC are also returned.

IMS 11 SPE User Exit Enhancements

- QUERY USEREXIT Command
  - SHOW() keywords

keyword	Output Returned
ALL	All possible output fields.
ACTIVE	The number of currently active instances of the user exit routine. This is a point in time number that represents the number of calls to the user exit that are still in progress and have not returned to IMS.
CALLS	The number of calls to the user exit since the user exit routine initial load.
ENTRYPT	The entry point address of the user exit routine.
ETIME	The total elapsed time (cumulative) in milliseconds spent in the exit module since user exit routine initial load.
LOADPT	The address at which the user exit routine was loaded.
RTIME	The local date and time that the user exit routine was initially loaded. The format of the output field is: yyyy-mm-dd hh:mm:ss.th
SIZE	The size in bytes of the user exit load routine. This value is displayed in hex.
TEXT	The 32 bytes starting from offset +04 from the exit module's entry point, translated to EBCDIC with non-printable characters replaced by periods (.).

The valid keywords that can be specified for SHOW() are the following:

ALL - All possible output fields are returned.

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

CALLS -The number of calls to the user exit since the user exit routine initial load. For performance reasons serialization is not obtained when IMS collects this number. For an exit type that can run multiple instances in parallel, this number is an approximation. The maximum value that can be displayed in this field is 2147483647 (2^31-1). If the call count exceeds this value, 2147483647 is displayed.

ENTRYPT - The entry point address of the user exit routine.

ETIME - The total elapsed time (cumulative) in milliseconds spent in the exit module since user exit routine initial load. For performance reasons serialization is not obtained when IMS collects this number. For an exit type that can run multiple instances in parallel, this number is an approximation. The maximum value that can be displayed in this field is 2147483647 (2^31-1). If the call count exceeds this value, 2147483647 is displayed

LOADPT - The address at which the user exit routine was loaded.

RTIME - The local date and time that the user exit routine was initially loaded. The format of the output field is: yyyy-mm-dd hh:mm:ss.th

SIZE - The size in bytes of the user exit load routine. This value is displayed in hexadecimal.

TEXT - The 32 bytes starting from offset +04 from the exit module's entry point, translated to EBCDIC with non-printable characters replaced by periods (.). This 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 displayed.

IMS 11 SPE User Exit Enhancements

- QUERY USEREXIT Example

TSO SPOC input:

```
QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)
```

TSO SPOC output:

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

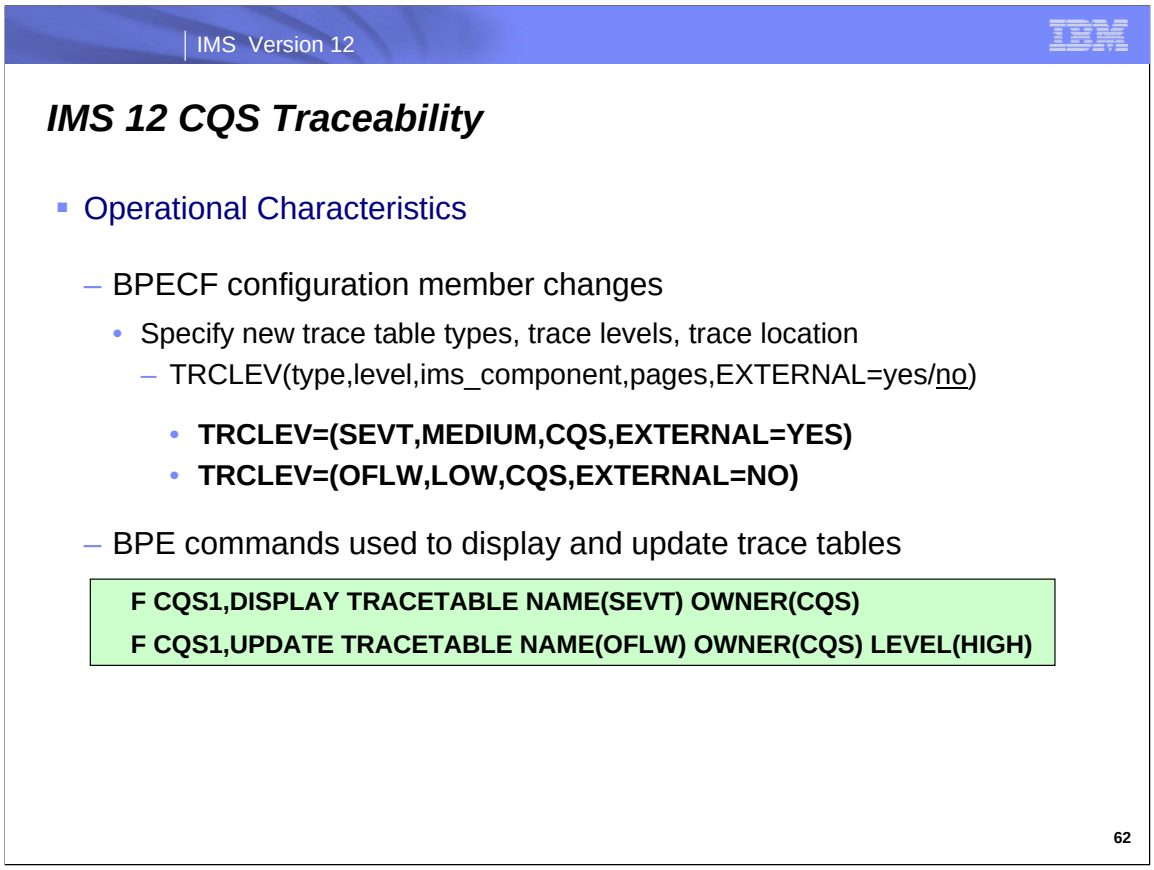

QUERY command input and resulting output display example:

QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)

For every INITTERM user exit which is currently supported, this command displays exit type, exit routine name, IMS member ID responsible for the output line, CC, number of calls to the exit and the local date and time when the exit was initially loaded.

Appendix

RAS Enhancements



The new CQS trace table specifications are defined in the BPE config proclib member. To enable BPE tracing so that it is always on for an IMS™ address space, you must define or modify the TRCLEV statements in the BPE configuration parameter member of the IMS.PROCLIB data set (specified by BPECFG=).

You can dynamically modify BPE tracing by using the z/OS MODIFY command with the UPDATE TRACETABLE command.

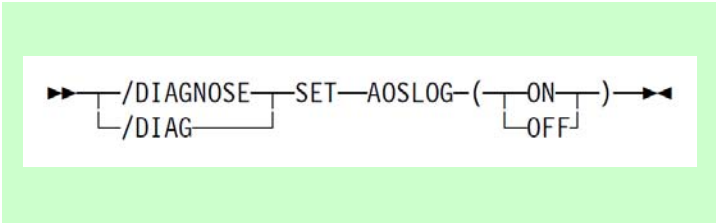
BPE trace records can be written to internal (memory only) trace tables and to external data sets. The default is to write to internal trace tables (EXTERNAL=NO). To write to an external data set, you must set EXTERNAL=YES on the TRCLEV statement and specify the EXTTRACE parameter in the BPE configuration parameter member. You must also define a generation data set group (GDG) for BPE to trace into. If you specify EXTERNAL=YES, trace data is written to both an external data set and internal trace tables.

You can dynamically change the external trace data set specification in the BPE configuration PROCLIB member and refresh the member while an address space is running. For example, if you are running without an external trace data set, you can edit your BPE PROCLIB member, add an external trace data set specification, and start using external trace without having to restart the address space.

You can display and update the CQS trace tables via commands.

New /DIAGNOSE Command SET AOSLOG Function

- /DIAGNOSE SET AOSLOG Command Syntax



  - SET
    - Specifies the attribute values to be changed
  - AOSLOG
    - Enables or disables logging of events related to APPC / OTMA synchronous transactions in a Shared Queues environment
    - If specified in a non-Shared Queues environment or when AOS=N in the DFSDCxxx proclib member ➔ command rejected with DFS2859I message

63

/DIAG SET AOSLOG() type-1 command syntax and keywords:

SET

Specifies the attribute values to be changed.

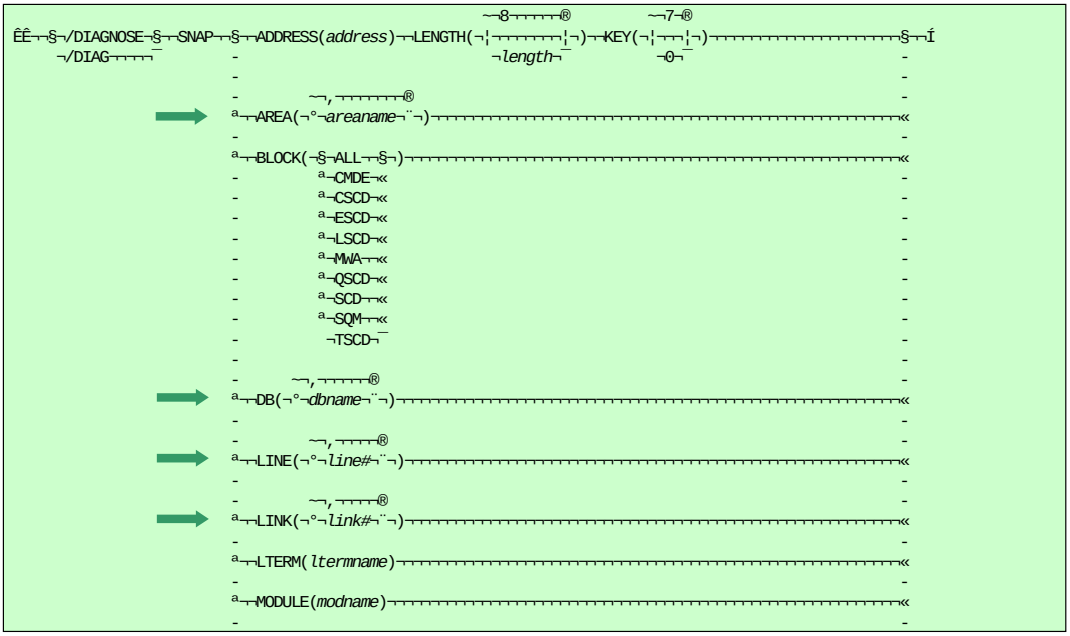
AOSLOG

Dynamically enables or disables logging of events related to APPC/OTMA synchronous transactions in a shared queues environment. When enabled the events are written to the IMS online log datasets (OLDS) as type X'6701' records.

If AOSLOG(ON) is specified in a non-shared queues environment or when AOS=N is specified in the DFSDCxxx proclib member, the command is rejected with DFS2859I message.

The SET AOSLOG() is added to the /DIAGNOSE command to enable or disable the logging of x'6701' records related to APPC/OTMA synchronous transactions in a shared queues environment.

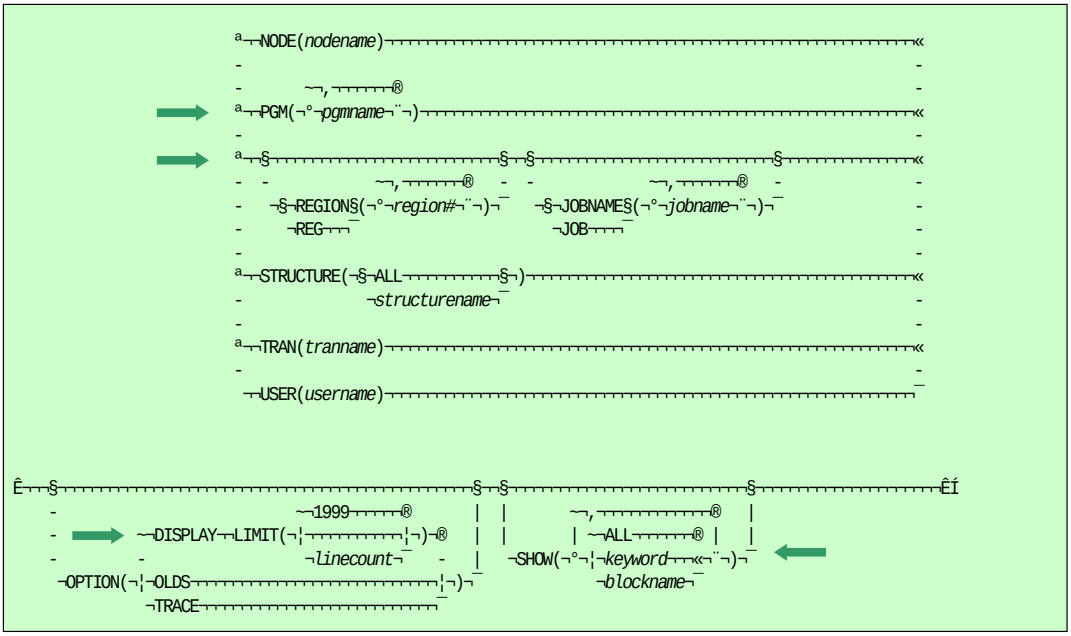
/DIAGNOSE SNAP Command Syntax …



The /DIAGNOSE command SNAP function is used to take a snap shot of system resources at any time without impacting IMS.

The /DIAGNOSE command syntax is shown with new resource types and new options identified.

/DIAGNOSE SNAP Command Syntax



The /DIAGNOSE command SNAP function is used to take a snap shot of system resources at any time without impacting IMS.

The /DIAGNOSE command syntax is shown with new resource types and new options identified.

New SNAP OPTION(DISPLAY) Option

- **DISPLAY** option added to specify output destination for captured data

```
/DIAGNOSE SNAP AREA(areaname) OPTION(DISPLAY)
```

  - Routes captured resource information back to issuing LTERM
    - MCS console, E-MCS console, MTO, SPOC, User Session, or AOI Program
  - Easier and quicker interactive capture of diagnostic data
    - No need to switch OLDS or stop TRACEs, and run batch extract to obtain snapped resource information
  - **DISPLAY** is the new default value for the OPTION parameter
    - **OLDS** ➔ SNAP output data is written to the active IMS log
    - **TRACE** ➔ SNAP output data is written to the active TRACE data set

The /DIAGNOSE command SNAP function OPTON has a new DISPLAY option.

DISPLAY: Specifies the destination for the resource information captured by the SNAP function.
The OPTION parameter is optional and has a default value of DISPLAY.
If DISPLAY is specified, SNAP data will be formatted and displayed on the issuing LTERM.
If OLDS is specified, SNAP data will be written to the IMS log.
If TRACE is specified, SNAP data will be written to the trace data sets.

New SNAP OPTION(DISPLAY) Option

- **LIMIT** option added to restrict amount of output data sent to LTERM

```
/DIAGNOSE SNAP DB(dbname) OPTION(DISPLAY,LIMIT(nnnn))
```

  - Optional linecount sub-parameter of the DISPLAY option
  - LIMIT default linecount is **1999** and valid linecount range is **1-9999**

The /DIAGNOSE command SNAP function OPTON has a new DISPLAY option.

LIMIT: Specifies a limit for the number of lines of formatted SNAP data to display in response to the command.
LIMIT is a sub-parameter of the OPTION parameter
The LIMIT parameter is optional and has a default value of 1999.
The linecount parameter must be numeric and in the range 1-9999.

New SNAP SHOW Option

- **SHOW** option added to control type and amount of output produced

```
/DIAGNOSE SNAP DB(dbname) SHOW(keyword/blockname)
```

  - Specifies the control block(s) to be captured by the SNAP command
  - Filtering values help limit command output to a manageable amount of data
    - Keywords
      - **ALL** - captures all control blocks that are available for specified resouce name(s)
      - **PRI** - captures the primary control blocks for specified resource name(s)
      - **OPT** - captures the optional control blocks that are available for specified resource name(s)
      - **APP** - captures the application control blocks that are available for REGION resource name(s)
      - **SYS** - captures the system control blocks that are available for REGION resource name(s)
    - Specific control block names for each resource type
  - Optional parameter of /DIAG SNAP command
    - If SHOW is omitted, only **PRI**mary control blocks for resource name(s) captured
  - Support for multiple SHOW() option parameters
    - Max number of parameters depends on the resource type

The /DIAGNOSE command SNAP function has a new SHOW option.

SHOW: Specifies the control blocks to be captured by the SNAP function.

The SHOW parameter is optional.

Valid filtering values for the keyword and blockname parameters are listed in slides describing values for each SNAP resource type.

SNAP Command Parameters

Resource Type	Resource Names	Resource Parameter Characteristics	SHOW ()	Max SHOW Options
AREA	areaname	Alphanumeric, <= 8 Characters	ALL, PRI, OPT and Control Blocks	16
DB	dbname	Alphanumeric, <= 8 Characters	ALL, PRI, OPT and Control Blocks	16
LINE	line#	Numeric, range of 1-1000	ALL, PRI, OPT and Control Blocks	16
LINK	link #	Numeric, range of 1-675	ALL, PRI, OPT and Control Blocks	16
PGM	pgmname	Alphanumeric, <= 8 Characters	ALL, PRI, OPT and Control Blocks	32
REGION	region#	Numeric, range of 1-999	ALL, PRI, OPT, SYS, APP and Control Blocks	64
JOBNAME	jobname	Alphanumeric, <= 8 Characters	ALL, PRI, OPT, SYS, APP and Control Blocks	64

69

The values for each resource type parameter must have certain attributes / characteristics (ie. alphanumeric, numeric, name length limitations, value ranges.)

The SHOW option keywords and control block names are specific for each resource type.

The number of SHOW option parameters depends on the individual resource type.

/DIAGNOSE SNAP Examples

/DIAGNOSE SNAP AREA(D0010001)

Response:

```
/DIAGNOSE SNAP STORAGE DISPLAY

Resource: AREA(D0010001)

ALDS     DEDB Area Name List Entry        Loc: 09E0A590
---- -------- -------- -------- --------  ----------------
0000 C4F0F0F1 F0F0F0F1 09BF7E90 09B26A30 |D0010001..=.....|

DMAC     DEDB Area Control Block          Loc: 09B26A30
---- -------- -------- -------- --------  ----------------
0000 F1F2F1F8 C4C5C4C2 D1F0F0F1 C4F0F0F1 |1218DEDBJ001D001|
0010 F0F0F0F1 0010213F 2217081F 00000000 |0001...........|
0020 00000000 000001EF 00008000 00000E5B |...............$|
0030 00000000 00000000 00000000 00000000 |................|
0040 00000000 00000000 00000000 00000000 |................|
0050 00000000 00000000 00000000 00000000 |................|
0060 00000000 00000000 00000403 00000000 |................|
0070 01C1000A 000F0019 00780002 00000200 |.A.............|
0080 000001F6 000003EA 00000005 00000000 |...6............|
0090 00000000 00018000 00000000 01000101 |................|
00A0 06088001 00000001 00000000 00000002 |................|
00B0 00000000 00000000 40000000 00000000 |........ .......|
 ...
04D0 00000000 00000000 00000000 00000000 |................|
04E0 00000000 00000000 00000000 00000000 |................|
04F0 00000000 00000000 00000000 00000000 |................|
0500 00000000 00000000                   |........        |
*10213/152244*
```


/DIAGNOSE SNAP Example

/DIAGNOSE SNAP AREA(D0010001)

Display primary fast path area control blocks (default)

/DIAGNOSE SNAP Examples

/DIAGNOSE SNAP AREA(D0010001) SHOW(DMAC) OPTION(DISPLAY,LIMIT(10))

Response:

```
/DIAGNOSE SNAP STORAGE DISPLAY

Resource: AREA(D0010001)

DMAC     DEDB Area Control Block          Loc: 09B26A30
---- -------- -------- -------- --------  ----------------
0000 F1F2F1F8 C4C5C4C2 D1F0F0F1 C4F0F0F1 |1218DEDBJ001D001|
0010 F0F0F0F1 0010213F 2217081F 00000000 |0001............|
0020 00000000 000001EF 00008000 00000E5B |...............$|
0030 00000000 00000000 00000000 00000000 |................|
*10213/152733*
```


/DIAGNOSE SNAP Example

/DIAGNOSE SNAP AREA(D0010001) SHOW(DMAC) OPTION(DISPLAY,LIMIT(10))

Display DMAC area control block for fast path area D0010001 and limit output back to requesting LTERM to 10 lines.

/DIAGNOSE SNAP Examples

/DIAGNOSE SNAP LINE(2,5) SHOW(CTT)

Response:

```
/DIAGNOSE SNAP STORAGE DISPLAY

Resource: LINE(2)

CTT      Communication Translate Table    Loc: 00055EF8
---- -------- -------- -------- --------  ----------------
0000 0F000000 00057D68 00000000 00082DA8 |......'........y|
0010 00000000 00000000 000A008F 67200000 |................|
0020 00000000 00840000 00000000 00000000 |.....d..........|

Resource: LINE(5)

CTT      Communication Translate Table    Loc: 00055EF8
---- -------- -------- -------- --------  ----------------
0000 0F000000 00057D68 00000000 00082DA8 |......'........y|
0010 00000000 00000000 000A008F 67200000 |................|
0020 00000000 00840000 00000000 00000000 |.....d..........|
*10208/063941*
```

72

/DIAGNOSE SNAP Example

/DIAGNOSE SNAP LINE(2,5) SHOW(CTT)

Display CTT control block for communication lines 2 and 5.

/DIAGNOSE SNAP Examples

/DIAGNOSE SNAP LINK(5,99,9) SHOW(CRB)

Response:

```
/DIAGNOSE SNAP STORAGE DISPLAY

Resource: LINK(5)

CRB      Communications Restart Block     Loc: 00055408
---- -------- -------- -------- --------  ----------------
0000 00000000 00000000 00000000 00008000 |................|
0010 00052568 00000000 00000000 00000000 |................|
0020 00000000 00000000 00000000 40404040 |............    |
0030 40404040 40404040 40404040 00000000 |            ....|
0040 00000000 00000000 00000000 00000000 |................|

Resource: LINK(99)

SNAP RESOURCE LINK(99) NOT FOUND

Resource: LINK(9)

CRB      Communications Restart Block     Loc: 00055548
---- -------- -------- -------- --------  ----------------
0000 00000000 00000000 00000000 00008000 |................|
0010 00052CE8 00000000 00000000 00000000 |...Y............|
0020 00000000 00000000 00000000 40404040 |............    |
0030 40404040 40404040 40404040 00000000 |            ....|
0040 D4E3D4E2 C3E5C1C2 00000000 00000000 |MTMSCVAB........|
*10190/112603*
```

73

/DIAGNOSE SNAP Example

/DIAGNOSE SNAP LINK(5,99,9) SHOW(CRB)

Display CRB control block for MSC logical links 5,99,9 – but 99 Is not a valid link.

/DIAGNOSE SNAP Examples

/DIAGNOSE SNAP PGM(STLDDLT1) SHOW(PDIR,RSCX)

Response:

```
/DIAGNOSE SNAP STORAGE DISPLAY

Resource: PGM(STLDDLT1)

PDIR     Program Directory Block (Master) Loc: 09C2F428
---- -------- -------- -------- --------  ----------------
0000 00000000 00000000 E2E3D3C4 C4D3E3F1 |........STLDDLT1|
0010 00000000 00370401 0215021B FFFF005C |..............*|
0020 80020000 09BBD9E8 00000000 09C2F480 |......RY.....B4.|
0030 004000AE 00000000 00000000 D7000000 |. ..........P...|
0040 00000000 00000000 00000000 00000000 |................|
0050 0C632508 00000000                   |........        |

RSCX     Resource Extension Block         Loc: 0C632508
---- -------- -------- -------- --------  ----------------
0000 C4C6E2D9 E2C3E740 00000060 00000000 |DFSRSCX ...-....|
0010 C65DAC37 00A1BFE0 00000000 00000000 |F).............|
0020 00000000 00000000 00000000 00000000 |................|
0030 40404040 40404040 00010002 00000000 |        ........|
0040 00000000 00000000 00000000 00000000 |................|
0050 00000000 00000000 00000000 00000000 |................|
*10213/184702*
```


/DIAGNOSE SNAP Example

/DIAGNOSE SNAP PGM(STLDDLT1) SHOW(PDIR,RSCX)

Display PDIR and RSCX control blocks for program STLDDLT1.

/DIAGNOSE SNAP Examples

/DIAGNOSE SNAP REG(1) SHOW(PST) OPTION(DISPLAY,LIMIT(10))

Response:

```
/DIAGNOSE SNAP STORAGE DISPLAY

Resource: REGION(1)

PST      Partition Specification Table    Loc: 09BEE060
---- -------- -------- -------- --------  ----------------
0000 80C4B32B 00000000 00000000 00000000 |.D..............|
0010 00000000 00000000 00000000 00000000 |................|
0020 09F6B040 00400038 00010018 40404040 |.6. . ......    |
0030 00000000 00000000 00000000 00000000 |................|
*10164/152614*
```


/DIAGNOSE SNAP Example

/DIAGNOSE SNAP REG(1) SHOW(PST) OPTION(DISPLAY,LIMIT(10))

Display PST control block for dependent region #1 and limit output back to requesting LTERM to 10 lines.

/DIAGNOSE SNAP Examples

/DIAGNOSE SNAP REG(1) JOBNAME(IMS1MPPB) SHOW(VTD)

Response:

```
/DIAGNOSE SNAP STORAGE DISPLAY

Resource: REGION(1)

VTD      SVC Vector Table Directory Entry Loc: 00FA8760
---- -------- -------- -------- --------  ----------------
0000 00FA8790 E2E8E2F3 00FAC700 04001210 |..g.SYS3..G.....|
0010 00FAC300 006C1448 C61D4F65 3DC92772 |..C..%..F.|..I..|
0020 006FF050 002A0000 09A660C0 00000000 |.?0&.....w-.....|

Resource: REGION(IMS1MPPB)

VTD      SVC Vector Table Directory Entry Loc: 00FA8790
---- -------- -------- -------- --------  ----------------
0000 00FA87C0 E2E8E2F3 00FAC580 04001210 |..g.SYS3..E.....|
0010 00FAC300 006C1448 C61D4F65 5870F5BA |..C..%..F.|...5.|
0020 006FF050 002B0000 09A66180 00000000 |.?0&.....w/.....|
*10163/165525*
```


/DIAGNOSE SNAP Example

/DIAGNOSE SNAP REG(1) JOBNAME(IMS1MPPB) SHOW(VTD)

Display VTD control block for dependent region #1 and dependent region with JOBNAME IMS1MPPB. (mixed resource name parameters – REG and JOBNAME)