Transactional VSAM: A System Programmer's Perspective

IBM zSeries Technical Conference

Session TSS04

Ruth Ferziger
ruthf@us.ibm.com

What is Transactional VSAM?

The objective of Transactional VSAM is to provide transactional recovery directly within VSAM. It is an extension to VSAM RLS. It allows any job or application that is designed for data sharing to read/write share VSAM recoverable files.

Transactional VSAM is a follow-on project/capability based on VSAM RLS (record level sharing). RLS provides a sysplex-wide server for sharing VSAM files. It provides CF (coupling facility) based locking and data caching with local buffer cross-invalidate. RLS supports CICS as a transaction manager. This provides sysplex data sharing of VSAM recoverable files when accessed through CICS. CICS provides the necessary unit-of-work management, undo/redo logging, and commit/back out functions. VSAM provides the underlying sysplex-scope locking and data access integrity.

Transactional VSAM adds logging and commit/back out support to VSAM RLS. Transactional VSAM requires/supports the OS/390 RRMS (Recoverable Resource Management Services) component as the commit or sync point manager.

Transactional VSAM provides a level of data sharing with built-in transactional recovery for VSAM recoverable files that is comparable to the data sharing and transactional recovery support for data bases provided by DB2 and IMSDB.

Transactional VSAM Overview

The transaction program execution model provides data sharing of recoverable resources. During the life of a transaction, its changes to recoverable resources are NOT seen by other transactions. The transaction may request that its changes be rolled back (backed out). If the transaction fails, its changes are backed out. This capability is called transactional recovery. It is provided by the resource managers.

Applications that are designed to the transaction model are able to easily share the recoverable resources. The resource managers provide the sharing isolation and recovery when a transaction fails or when the execution environment fails.

IMSDB and DB2 are resource managers that provide transactional recovery for their databases. CICS File Control provides transactional recovery for VSAM recoverable files. Now, Transactional VSAM provides transactional recovery for VSAM recoverable files as well.

Transactional VSAM Logging

Transactional VSAM logging uses the OS/390 System Logger. The Transactional VSAM logger is a reuse of the design and much of the code of the CICS logger. Forward recovery logstreams for VSAM recoverable files will be shared across CICS and Transactional VSAM. The forward recovery log stream is specified as an attribute of the file. It is specified via AMS. The parameter is: LOGSTREAMID=name.

CICS/Transactional VSAM Logstreams

Transactional VSAM performs the logging for VSAM RLS data sets accessed in Transactional VSAM mode. You can share a forward recovery log stream between multiple data sets - you do not need to define a log stream for each forward-recoverable data set. Your decision is a balance of transaction performance, rapid recovery, and the work involved in managing a large number of log streams.

The MVS logger merges all the forward recovery log records from the various Transactional VSAM instances onto the shared forward recovery log. Some points to consider are:

1. All data sets used by one transaction should use the same log stream (to reduce the number of log streams written to at sync point).
2. Share a forward recovery log stream between data sets that:
   - Have similar security requirements
   - Have similar backup frequency
   - Are likely to need restoring in their entirety at the same time.
3. Log stream names should relate to the data sets. For example, PAYROLL.data_sets could be mapped to a forward recovery log named PAYROLL.FWDRECOV.PAYLOG.
4. Don't mix high update frequency data sets with low update frequency data sets, because this causes a disproportionate amount of unwanted log data to be read during recovery of low frequency data sets.
5. Don't put all high update frequency data sets on a single log stream because you could exceed the throughput capacity of the stream.
6. If you define too many data sets to a single log stream, you could experience frequent structure-full events when the log stream can't keep up with data flow.

7. Redundant data should be deleted from log streams periodically so that the log streams do not become excessively large. Typically, for a forward recovery log, deletion of old data is related to the data backup frequency. For example, you might keep the 4 most recent generations of backup, and when you delete a redundant backup generation you should also delete the corresponding redundant forward recovery log records. These are the records older than the redundant backup because they are no longer needed for forward recovery.

The log of logs is written to provide information to forward recovery programs such as CICS VSAM Recovery (CICSVR) The log of logs contains copies of the start of run records, and the tie-up and file close records for forward recoverable data sets written to forward recovery logs. Thus it provides a summary of which recoverable VSAM data sets Transactional VSAM has used, when they were used, and to which log stream the forward recovery log records were written.

If you have a forward recovery product that can utilize the log of logs, you should ensure that all Transactional VSAM instances sharing the recoverable data sets write to the same log of logs log stream. Do not share the log of logs between test and production Transactional VSAM instances, because it could be misused to compromise the contents of production data sets during a restore.

Transactional VSAM & z/OS RRS

Transactional VSAM is a recoverable resource manager. It is NOT a commit or sync point manager. Transactional VSAM interfaces with the OS/390 Sync point Manager. When an application issues a commit request directly to OS/390 or indirectly through a sync point manager that interfaces with the OS/390 sync point manager, Transactional VSAM is invoked to participate in the 2-phase commit process. Other resource managers (like DB2) whose recoverable resources were modified by the transaction are also invoked by the OS/390 sync point manager thus providing a commit scope across the multiple resource managers.

Define Log Structures

You must define the coupling facility structures needed for all the log streams that are used by your Transactional VSAM instances. You define structures in the MVS system logger LOGR policy in the system logger couple data sets using the DEFINE STRUCTURE specification of the ICXMIAPU utility.

Coupling facility space is divided into structures using the coupling facility resource management (CFRM) policy. The CFRM policy allows you to define how MVS is to manage coupling facility resources.

Multiple log streams can write data to a single coupling facility
structure. This does not mean that the log data is merged; the log data stays segregated according to log stream. You can specify the number of log streams that use the resources of a single

coupling facility structure using the LOGSNUM parameter on the IXCMIAPU service to define a structure.

Each log stream is allocated a proportion of the structure space based on the number of currently connected log streams (up to the limit specified in LOGSNUM). For example, a structure may be defined to contain a maximum of, say, 30 log streams. If only 10 log streams are connected, each log stream can use one tenth of the space in the structure. As other log streams are connected and disconnected, the MVS system logger adjusts the proportion of space to be used by each log stream.

It is important to plan carefully before specifying a value for LOGSNUM, because this parameter determines how much storage space in the structure is available to each log stream. A number in the range 10 to 20 is optimum in many environments.

Bear in mind the following points when planning the definition of your coupling facility structures:

1. The CFRM policy allows a maximum of 128 structures for all purposes.
2. Allow 10 - 20 log streams per structure unless this would require too many structures.
3. Smaller structures are more quickly allocated, rebuilt, and recovered than larger ones
4. It is good practice to keep the log streams for testing Transactional VSAM in structures separate from the structures holding the log streams of production Transactional VSAM instances. This prevents the structure space available to production Transactional VSAM instances being affected by structure usage of the test Transactional VSAM instances.
5. Share structures between MVS images. If an MVS image or logger address space fails, and a surviving MVS image is using the same log stream structures (although not necessarily the same log streams), the surviving image is notified of the failure and can initiate immediate log stream recovery for the failing MVS. Recovery would, otherwise, be delayed until the next time that a system attempts to connect to a log stream in the affected structures, or until the logger address space of the failing system is restarted.
6. Use the appropriate buffer size. The average buffer size defined for a structure should be reasonably close to the actual buffer size of the log streams using the structure. If it is not, there is a risk that usable space will be exhausted long before the structure is actually full.
7. Set MAXBUFSIZE to slightly less than 64K - say, 64000. This allows Transactional VSAM to write the maximum size user record (60K) and allows coupling facility storage to be allocated in 256-byte units. There is no significant advantage in setting MAXBUFSIZE lower than 64000 as far as the utilization of storage is concerned. If you allow MAXBUFSIZE to default, coupling facility storage is allocated in 512-byte units. This can be wasteful of storage.
8. Set a low value for the REBUILDPERCENT parameter in the CFRM policy for log structures used for system logs.

Define Logstreams

The JCL shown defines log streams that Transactional

VSAM will use. It is meant for guidance only and you should substitute values appropriate to your requirements.

In this example, each of the log streams has its own structure. This is just an example. Your log streams may share structures.

It is strongly recommended that you specify DIAG(YES) for the Transactional VSAM system logs. This parameter indicates special Logger diagnostic activity is allowed for this logstream. This is especially useful when certain Logger errors occur, such as X'804', that indicate that some data may have been lost.

Important note: Log stream and log stream staging data sets are single extent VSAM linear data sets and need shareoptions(3,3). The default is shareoptions(1,3), so you must specify shareoptions explicitly.

RACF Definitions for Transactional VSAM Logstreams

Installations must define authorization to system logger resources in order for Transactional VSAM to be able to access, read and write its log streams (undo, shunt, and forward recovery log streams). This applies to both coupling facility and DASD-only log streams. It is recommended that the RLS server address space (SMSVSAM) be assigned privileged and/or trusted RACF status. Privileged means that most RACROUTE REQUEST=AUTHs done for SMSVSAM are considered successful, without actually performing any checking. In addition, in these cases RACF:

1. does not call any exit routines
2. does not generate any SMF records
3. does not update any statistics

This bypassing also applies to the checking done for the CHKAUTH operand on the RACROUTE REQUST=DEFINE macro instruction. All other RACF processing occurs as usual.

Trusted is similar and also means that most RACROUTE REQUEST=AUTHs done for SMSVSAM are considered successful, without actually performing any checking. In addition, in these cases RACF:

1. does not call any exit routines
2. does generate SMF records based on the audit options specified in SETROPTS LOGOPTIONS and the UAUDIT setting in the userid profile
3. does not update any statistics

This bypassing also applies to the checking done for the CHKAUTH operand on the RACROUTE REQUST=DEFINE macro instruction. All other RACF processing occurs as usual.

For more information see "OS/390 Security Server (RACF) System Programmer's Guide" (SC28-1913).

If the RLS server address space is neither privileged nor trusted, SMSVSAM must be granted the appropriate access authorization to log stream profiles in the RACF LOGSTRM general resource class to the userid of the RLS address space, SMSVSAM. You must first associate SMSVSAM with a RACF defined userid, for example, SYSTASK, in the started procedures table. It is recommended that you do not use a userid of SMSVSAM as this does not always work. Please see OS/390: Security Server (RACF) System Programmer's Guide, "Associating Started Procedures and Jobs with User IDs," (SC26-1913) for more information.

There is no facility within Transactional VSAM for controlling LOGSTRM security checks. This is controlled by the MVS security administrator activating the LOGSTRM general resource class by means of the SETROPTS command. If the LOGSTRM resource class is active, then Transactional VSAM needs UPDATE authority to the log stream profiles for the logs to which it writes, granted using the userid of the RLS address space, SMSVSAM. The log streams need to be defined to MVS.

Permit READ access to those users who need to read the Transactional VSAM log streams.

The generic profile in the following example shown could be defined to cover all the log streams referenced by a Transactional VSAM instance.

The examples that follow give access to three categories of users.

In these examples, smsvsam_userid is the userid of the RLS address space which is the address space in which Transactional VSAM makes its calls to the MVS Logger. The number of profiles you define depends on the naming conventions of the logs, and to what extent you can use generic profiling.

RACF Definitions for Forward Recovery Logstreams

Transactional VSAM also writes to forward recovery log streams and a log of logs which is used to optimize forward recovery. To protect these log streams, appropriate RDEFINEs and PERMITs should be done for each of them. Transactional VSAM must be granted access to the log of logs and the forward recovery log streams for all forward recoverable data sets which may be accessed in Transactional VSAM mode. A PERMIT granting UPDATE authority to each of these log streams is required.

RACF Definitions for Peer Recovery

In order for peer recovery to be possible, the RLS server must also be granted authority to read and write to the log streams of the Transactional VSAM instances on other systems. For example, if Transactional VSAM instance IGWTV001 running on SYSTEM1 wanted to

perform peer recovery for Transactional VSAM instance IGWTV002 which had been running on SYSTEM2, it would need authority to its own log streams and well as those of IGWTV002.

SMS Constructs

Transactional VSAM does not provide any mechanism for the automatic deletion of records from forward recovery log streams. It is your responsibility to manage the size of these log streams. If you want to keep them from becoming excessively large and you need long-term data retention, then you may want to copy the data from log stream storage into alternative archive storage.

The MVS system logger supports an option that manages the migration of log data from data sets to other media, such as tape, using DFSMShsm. This data is always retrievable online.

The MVS system logger automatically maintains two copies of log data by using:

1. A coupling facility and a data space, or
2. A coupling facility and staging data sets.

When a coupling facility log stream becomes full, the older records are moved onto SMS-managed data sets. When the log data is on DASD, high data availability can be assured by:

1. The dual copy facilities of the 3990 control unit, which can be used to maintain a second copy of the data, or
2. The RAID 5 fault-tolerant storage facilities provided by the IBM RAMAC Array Subsystem and the IBM RAMAC Array DASD.

SYS1.PARMLIB Changes

Some Transactional VSAM parameters apply only to the system on which they are found. Others apply across the sysplex. Regardless of which type a parameter may be, values are not remembered across IPLs. Therefore, your IGDSMSxx member of SYS1.PARMLIB must always specify a complete set of the parameters you wish Transactional VSAM to use.

The RLSTMOUT parameter is optional. It specifies the maximum time in seconds that a VSAM RLS or Transactional VSAM request is to wait for a required lock before the request is assumed to be in deadlock and aborted with VSAM return code 8 and reason code 22(X'16'). RLSTMOUT is specified as a value in seconds in the range of 0 to 9999. The default is 0. A value of 0 means that the VSAM RLS or Transactional VSAM request has no time out value; the request will wait for as long as necessary to obtain the required lock.

VSAM RLS detects deadlocks within VSAM and Transactional VSAM. It cannot detect deadlocks across other resource managers, and uses the time out value to determine when such

deadlocks may have occurred. The installation may specify a global time out value in the IGDSMSxx member of SYS1.PARMLIB, a step level time out value on the JCL, and specific applications may specify a time out value on the RPL passed for each VSAM request.

For a particular VSAM RLS or Transactional VSAM request, the value used for time out is:

1. the value specified in the RPL, if any
2. the value specified in JCL at the step level, if any
3. the value specified in the IGDSMSxx member of SYS1.PARMLIB, if any

RLSTMOUT is both a VSAM RLS and a Transactional VSAM parameter. Therefore if RLSTMOUT is found but no Transactional VSAM instance names are specified, then the value is used only by RLS. RLSTMOUT may be specified only once in a sysplex and applies across all systems in the sysplex. The first instance of Transactional VSAM brought up within the sysplex determines the value. Subsequent Transactional VSAM instances use the value established by the first system, regardless of what may be specified in their members of SYS1.PARMLIB. To change the value, use the SETSMS command. This will cause the value to be changed on all systems in the sysplex.

The SYSNAME parameter is optional. It specifies the name of the systems to which the preceding or subsequent Transactional VSAM instance names apply. Up to 32 system names may be specified. The system names must be specified in the same order as the Transactional VSAM instance names. SMS examines the system names specified and compares them to the system name in the CVT. When a match is found, it stores the value of the TVSNAME parameter in the matching position as the Transactional VSAM instance name for the system. The combination of SYSNAME. and TVSNAME should be used when the PARMLIB member is shared between systems.

If no SYSNAME. parameter is specified, then the TVSNAME applies to the system on which the PARMLIB member is read. This parameter is supported only in PARMLIB. It is not supported on the SETSMS command.

If SYSNAME. is found without TVSNAME, it is treated as a syntax error. If SYSNAME. is found, but the system is not listed, then Transactional VSAM will not be started on the system.

The TVSNAME parameter is optional. It specifies the identifiers which uniquely identify instances of Transactional VSAM running in the sysplex. Up to 32 identifiers may be specified. The identifiers must be unique within the sysplex. They must be a numeric value from 0 to 255, which Transactional VSAM uses as the last byte of its instance name (although it will be displayed as three bytes).

If the TVSNAME parameter is specified without the SYSNAME. parameter, then only a single value may be specified, and the name applies to the system on which the SYS1.PARMLIB member is read. The numeric digits are appended to the characters IGWTV to form the Transactional VSAM instance name. The TVSNAME parameter may be used without the

SYSNAME. parameter when the PARMLIB member is not shared between systems. This parameter is supported only in PARMLIB. It is not supported on the SETSMS command, and there is no default.

If no TVSNAME parameter is found, then Transactional VSAM processing will not be available on the system.

The TV_START_TYPE parameter is optional. It specifies the type of start Transactional VSAM is to perform. Up to 32 TV_START_TYPE values may be specified. TV_START_TYPE values must be specified in the same order as Transactional VSAM instance names. If WARM is specified, then Transactional VSAM reads its undo log and processes the information it finds in accordance with the information RRS has about any outstanding URs. If COLD is specified, then Transactional VSAM deletes any information remaining in the undo log and starts as if the log were empty. COLD should be used when the Transactional VSAM undo log has been damaged. The default is WARM.

You can allow some values to default and others to be specified by entering something like:

TV_START_TYPE(COLD,,COLD)

This would cause the Transactional VSAM instance in the second position to warm start, while the first and third Transactional VSAM instances would cold start.

If TV_START_TYPE is found with only the TVSNAME parameter, then it applies to the system only which the SYS1.PARMLIB member is being read. If it is found with both TVSNAME and SYSNAME parameters, then it applies to the system specified on the SYSNAME parameter preceding or following it. If the TVSNAME parameter is not specified and TV_START_TYPE is, it is treated as a syntax error.

This parameter is supported only in SYS1.PARMLIB. It is not supported on the SETSMS command.

The AKP parameter is optional. It specifies the activity key point trigger value, which is the number of logging operations between the taking of key points. Up to 32 activity key point values may be specified. AKP values must be specified in the same order as Transactional VSAM instance names. Valid values are 200 to 65535. The default is 1000.

You can allow some values to default and others to be specified by entering something like:

AKP(800,,3000)

This would cause the value for the system/Transactional VSAM instance in the second position to default to 1000, and set the values for the first and third systems/Transactional VSAM instances to 800 and 3000 respectively.

If AKP is found with only the TVSNAME parameter, then it applies to the system only which the SYS1.PARMLIB member is being read.

If AKP is found without TVSNAME, it is treated as a syntax error.

The log of logs parameter is optional. It specifies the log stream to be used as the log of logs. This log contains copies of the tie up records written to forward recovery logs and is used by forward recovery products. If it is not specified, then no log of logs is used. The default is to use no log of logs.

If LOG_OF_LOGS is found but no Transactional VSAM instance names are specified, then it will be treated as a syntax error.

This parameter is unique to each system in the sysplex. As each instance of Transactional VSAM comes up, it uses the log of logs name found in its member of SYS1.PARMLIB.

If you are running test and production systems within the same sysplex, it may be desirable to use different logs of logs. In this case, use a separate SYS1.PARMLIB member for the test system and specify a different log of logs in it.

This parameter is supported only in SYS1.PARMLIB. It is not supported on the SETSMS command.

The QTIMEOUT parameter is optional. It specifies the quiesce exit timeout value in seconds. Only one quiesce timeout value may be specified and applies to all systems in the sysplex. The first instance of Transactional VSAM brought up within the sysplex determines the value. Subsequent Transactional VSAM instances use the value established by the first system, regardless of what may be specified in their members of SYS1.PARMLIB. To change the value, use the SETSMS command. This will cause the value to be changed on all systems in the sysplex.

The quiesce timeout value specifies the amount of time the Transactional VSAM quiesce exits will allow to elapse before concluding that a quiesce cannot be completed successfully. Valid values are 60 to 3600. The default is 300.

If QTIMEOUT is found but no Transactional VSAM instance names are specified, then it is treated as a syntax error. Unlike other Transactional VSAM parameters, it is specified only once and applies across all systems.

The MAXLOCKS parameter is optional. It specifies two values: the maximum number of unique lock requests that a single unit of recovery may make before warning messages are issued and an increment value. Once the maximum number of unique lock requests is reached, the warning messages will be issued every time the number of unique lock requests over and above the maximum increases by a multiple of the increment. The messages include the job name of the job which is holding the locks. The installation can then determine if the application should be

allowed to proceed or if the job should be canceled. If the job is canceled, the UR will be backed out, and the locks will remain held until the back out completes.

MAXLOCKS is specified as a pair of values in the range of 0 to 999999. The default for both values is 0. A value of 0 indicates that warning messages should not be issued.

If MAXLOCKS is found but no Transactional VSAM instance names are specified, then it will be treated as a syntax error. Unlike other Transactional VSAM parameters, it is specified only once and applies across all systems.

SYS1.PARMLIB Rules

The following rules apply to the Transactional VSAM parameters:

1. If any Transactional VSAM parameter is specified but TVSNAME is not, it is treated as a syntax error.

2. TVSNAME may be specified without SYSNAME. only when there is only a single TVSNAME specified, for example TVSNAME(1).

3. If more than one TVSNAME value is specified and SYSNAME. is not, it is treated as a syntax error.

4. All Transactional VSAM parameters which take multiple values must either accept defaults for all values or specify the same number of values. That is, AKP specifying 2 values and TV_START_TYPE specifying 3 values is invalid.

5. If Transactional VSAM parameters are found in the IGDSMSxx member of SYS1.PARMLIB by a system which is not listed in SYSNAME. then Transactional VSAM will not be started on that system. (The Transactional VSAM parameter will be ignored.)

6. Syntax errors such as TVSNAME(1,2,3) SYSNAME.(SYS1,SYS2,SYS3) TV_START_TYPE(COLD,WARME,COLD) will be flagged only on the system to which they apply, in the case, SYS2. Note, however, that errors which the parser cannot handle such as (COLD,WARMEST,COLD) may be treated as an error on all systems. In this case, the parser disallows the specification because the value specified exceeds the maximum length the parser was told to accept.

7. Syntax errors where there is a mismatch in the number of values specified, such as TVSNAME(1,2) SYSNAME.(SYS1,SYS2,SYS3), will be flagged on all systems, because Transactional VSAM is unable to determine what systems the TVSNAMEs were intended to apply.

8. If there was a TVSNAME specified on the system, but an IGDSMSxx member of SYS1.PARMLIB is subsequently activated which does not specify TVSNAME, the

TVSNAME will be cleared. If Transactional VSAM comes down (or the SMSVSAM server recycles), Transactional VSAM will not be restarted.

About the MAXLOCKS Parameter

This parameter specifies two values: the maximum number of unique lock requests that a single unit of recovery may make before warning message IGW859I is issued to the system console and message IGW10074I is issued to the job log, and an increment value. Once the maximum number of unique lock requests is reached, the warning messages will be issued every time the number of unique lock requests over and above the maximum increases by a multiple of the increment. The messages include the job name of the job which is holding the locks. The installation can then determine if the application should be allowed to proceed or if the job should be canceled. If the job is canceled, the UR will be backed out, and the locks will remain held until the back out completes.

Lock requests are considered unique if they lock different records within the base cluster. Repeated requests for the same base cluster records will not result in the count being incremented.

Warning messages IGW859I and IGW10074I are not issued for units of recovery that are in back out. This is because a unit of recovery which is in back out cannot obtain locks on any additional records. In addition, the only actions that can be taken are to allow the job to continue or to cancel the job. Since canceling the job would result in back out, issuing the messages for units of recovery which are already in back out serves no purpose.

Messages IGW859I and IGW10074I are issued until the unit of recovery reaches commit. Once the unit of recovery enters commit, no additional messages will be issued. This is because the unit of recovery can no longer lock any additional records, and the number of locks it holds will not change until the commit is completed, at which point all those locks that can be released will be. Any locks which cannot be released will become retained.

To avoid flooding the system console with messages, messages IGW859I and IGW10074I are issued by an asynchronous timer driven task which wakes up every 10 seconds. This means that the messages will not necessarily reflect the exact values specified for the maximum and the increment, but rather will reflect the values which represent the state of the unit of recovery at the time the task awakens.

Each time it wakes up, it scans the units of recovery that are in progress for any that have exceeded the maximum or which had previously exceeded the maximum and have now exceeded the increment value as well. If it is necessary to issue a new message for a unit of recovery for which a message had previously been issued, the old message will be DOMed from the system console and replaced.

MAXLOCKS takes into account the number of unique lock requests. It does not count the actual number of locks obtained. The number of locks requested will differ from the number of locks

held when alternate indexes are used. If an update modifies alternate keys, then a lock is obtained for the base record, for each old alternate key, and for each new alternate key. Therefore, if n alternate keys are modified, a single lock request can result in obtaining (2n+1) locks.

MAXLOCKS is specified as a pair of values in the range of 0 to 999999. The default for both values is 0. A value of 0 indicates that warning messages IGW859I and IGW10074I should not be issued.

If MAXLOCKS is specified but Transactional VSAM is not active, it has not effect. The MAXLOCKS values are sysplex-wide. Changing them on one system changes them on all systems in the sysplex.

1. Lock requests are considered unique if they lock different records within the base cluster. Repeated requests for the same base cluster records will not result in the count being incremented.

2. The warning messages are not issued for units of recovery that are in back out. This is because a unit of recovery which is in back out cannot obtain locks on any additional records. In addition, the only actions that can be taken are to allow the job to continue or to cancel the job. Since canceling the job would result in back out, issuing the messages for units of recovery which are already in back out serves no purpose.

3. The warning messages are issued until the unit of recovery reaches commit. Once the unit of recovery enters commit, no additional messages will be issued. This is because the unit of recovery can no longer lock any additional records, and the number of locks it holds will not change until the commit is completed, at which point all those locks that can be released will be. Any locks which cannot be released will become retained.

4. To avoid flooding the system console with messages, messages, the warning messages are issued by an asynchronous timer driven task which wakes up every 10 seconds. This means that the messages will not necessarily reflect the exact values specified for the maximum and the increment, but rather will reflect the values which represent the state of the unit of recovery at the time the task awakens.

   Each time it wakes up, it scans the units of recovery that are in progress for any that have exceeded the maximum or which had previously exceeded the maximum and have now exceeded the increment value as well. If it is necessary to issue a new message for a unit of recovery for which a message had previously been issued, the old message will be DOMed from the system console and replaced.

5. MAXLOCKS takes into account the number of unique lock requests. It does not count the actual number of locks obtained. The number of locks requested will differ from the number of locks held when alternate indexes are used. If an update modifies alternate keys, then a lock is obtained for the base record, for each old alternate key, and for each new alternate

key. Therefore, if n alternate keys are modified, a single lock request can result in obtaining (2n+1) locks.

DISPLAY SMS Command

DISPLAY SMS,TRANVSAM

This command is used to display information about the instance of Transactional VSAM on this system or on all systems in the sysplex, when the ALL keyword is specified. The output includes:

- The activity key point (AKP) trigger, which is the number of logging operations between the taking of keypoints.
- The status of this instance of Transactional VSAM (active, quiescing, starting, or inactive)
- How Transactional VSAM started (cold - log data not read, and any old data discarded; or warm - log data read and processed)

- Transactional VSAM status with respect to RRS
- The quiesce timeout value
- The log of logs, if one is in use
- The number of active units of recovery (URs)

Message IGW800I is issued in response to this command.

DISPLAY SMS,JOB

This command is used to display information about a particular job which is using Transactional VSAM services on one of the systems in the sysplex. The output includes:

- The name of the current step within the job
- The current URID for the job
- The status of the unit of recovery (in-reset, in-flight, in-prepare, in-commit, in-backout, indoubt)

Message IGW801I is issued in response to this command.

DISPLAY SMS,URID

This command is used to display information about a particular unit of recovery (UR) currently active within the sysplex or about all units of recovery currently active on the system on which the command was issued on whose behalf Transactional VSAM has performed any work. It does not include information about work which has been shunted, since that information can be displayed using the DISPLAY SMS,SHUNTED command. It also does not include information

about URs which may be in restart processing as a result of an earlier failure. This work is not considered to be currently active, since it is not associated with any batch job, and the units of recovery associated with it will terminate as soon as commit or back out processing for them can be completed. The output includes:

- The age of the unit of recovery
- The jobname with which the unit of recovery is associated
- The name of the current step within the job
- The status of the unit of recovery (in-reset, in-flight, in-prepare, in-commit, in-backout, indoubt)
- The userid associated with the job

Message IGW802I is issued in response to this command.

DISPLAY SMS,SHUNTED

This command is used to display the entries currently contained in the shunt logs of the systems in the sysplex. Entries are moved to the shunt log when Transactional VSAM is unable to finish processing a sync point for them, for example, due to an I/O error. As long as a shunted entry exists, the locks associated with that entry are retained. The following is a list of reasons why a unit of recovery might be shunted. Some require reallocating the data set. For information on moving and reallocating data sets, see the section on moving data sets in the CICS Recovery and Restart Guide.

- A VSAM RLS cache structure, or connection to it, has failed.
- A logstream became or was made unavailable.
- A sync point for a unit of recovery failed for one of the data sets it was using. In this case, the name of the data set and the reason for the failure are included. Failure reasons include:

  • Commit failed.
  • An error occurred at some point when RLS locks were in the process of being released. This is an error that can normally be resolved by recycling the SMSVSAM server (which should happen automatically).
  • The locks were acquired as a result of recoverable requests having been issued against the data set.
  • A data set is full; no space is available on the direct access device for adding records to it. You need to reallocate the data set with more space. You can then retry the back out using SHCDS RETRY.
  • Back out failed. This occurs as a result of a severe error being identified during back out, and is possibly an error in either Transactional VSAM or VSAM. The problem may go away if the back out is retried.
  • Index record is full. A larger alternate index record size needs to be defined for the data set.
  • A hard I/O error occurred during back out. To correct this error, restore a full backup copy of the data set and perform forward recovery.

- An attempt to acquire a lock during back out of an update to this data set failed because the RLS lock structure was full. You must allocate a larger lock structure in an available coupling facility and rebuild the existing lock structure into it, then use the SHCDS RETRY command to drive the back out retry.
- Error on opening the data set for back out. A console message notifies you of the reason for the open error. One likely reason could be that the data set was quiesced.

Message IGW803I is issued in response to this command.

There are three types of information that can be displayed in response to this command:

1. When neither the SPHERE nor UR keyword is specified, this command results in a list of systems in the sysplex and the number of units of recovery which that system has shunted.
2. When the SPHERE keyword is specified, this command results in a list of shunted work for the sphere specified for all of the systems in the sysplex.
3. When the UR keyword is specified, this command results in a list of shunted work for the unit of recovery specified for all of the systems in the sysplex. When ALL is specified, this command results in a list of shunted work for all shunted units of recovery for all the systems in the sysplex. To avoid flooding the console, Transactional VSAM will write out 255 lines and then issue a WTOR to determine whether or not to continue.

If the error is correctable, the installation may choose to fix the problem and then request that Transactional VSAM again attempt processing of the entry by issuing the SHCDS RETRY command. If the data set cannot be restored to a point where it is consistent with the log entry so that it does not make sense to attempt processing of the log entry again, the installation may choose to discard the log entry by issuing the SHCDS PURGE command.

DISPLAY SMS,LOG

This command displays information about a logstreams that Transactional VSAM is currently using on one of the systems in the sysplex. If ALL is specified, it displays information about all of the log streams in use on the system on which the command is issued. The output includes the status of the log stream (failed or available), type of log (undo, shunt, forward recovery, or log of logs), the job name and URID of the oldest unit of recovery using the log, and a list of all Transactional VSAM instances that are using the log. If information about a specific log stream is requested and the log stream is either a system log or a forward recovery log, the output includes the names of the jobs using the log stream.

This command might be issued to determine why a logstream is increasing in size. If a unit of recovery (UR) is long-running, then Transactional VSAM would be unable to delete any log blocks containing data associated with it, which in turn would make it impossible to truncate the logstream.

Message IGW804I is issued in response to this command.

DISPLAY SMS,DSNAME

For a given fully qualified data set name, this command displays the jobs currently accessing the data set using Transactional VSAM access on the systems within the sysplex. Message IGW805I is issued in response to this command.

DISPLAY SMS,OPTIONS

When Transactional VSAM is not running on the system, the output of this command is unchanged. When Transactional VSAM is running on the system, the output of this command include Transactional VSAM related information.

VARY SMS Command

VARY SMS,TRANVSAM

This command is used to enable, quiesce, or disable the specified Transactional VSAM instance, or all Transactional VSAM instances in the sysplex. It is routed to all systems in the sysplex and affects either all Transactional VSAM instances or the Transactional VSAM instance with the name specified.

Messages IGW471I and IGW472I are issued in response to this command.

QUIESCE: Transactional VSAM should complete processing of any in progress units of recovery but not accept any new ones. Transactional VSAM will complete its quiesce processing when the last data set which is open for Transactional VSAM access is closed. Transactional VSAM will be unavailable until a VARY SMS,TRANVSAM,ENABLE command is issued.

DISABLE: Transactional VSAM immediately stops processing new work requests, including units of recovery which are currently in progress. When the last data set which is open for Transactional VSAM access is closed, Transactional VSAM will retain locks, unregister with RRS, and will be unavailable until a VARY SMS,TRANVSAM,ENABLE command is issued. No further Transactional VSAM requests can complete until Transactional VSAM is enabled, including commit and back out requests.

ENABLE: Transactional VSAM should begin accepting new units of recovery for processing.

VARY SMS,LOG

This command is used to enable, quiesce, or disable Transactional VSAM access to the specified logstream. Quiescing the Transactional VSAM undo or shunt logstream is equivalent to quiescing Transactional VSAM processing. Disabling the Transactional VSAM undo or shunt logstream is equivalent to disabling Transactional VSAM. Although the two logstreams are physically separate, they are treated as a single entity by Transactional VSAM logging services.

Quiescing or disabling the log of logs has no effect on Transactional VSAM processing since records are written to the log of logs only as an optimization for forward recovery products. However, disabling the log of logs may cause a mismatch of tie up records written at data set open with file close records.

Quiescing a Data Set

Unlike CICS, Transactional VSAM does not control the opens and closes of data sets accessed using it. Because of this, Transactional VSAM supports a limited form of QUICLOSE. If the data set is not currently open for Transactional VSAM access, the quiesce is accepted, otherwise it is rejected. Therefore, in order to completely quiesce a data set, you must first ensure that it is not currently in use using Transactional VSAM access.

The first step is to determine which jobs are using the data set. You can do this either using the DISPLAY SMS,DSNAME
operator command or the IDCAMS SHCDS LISTDS(dsname),JOBS command.

Next, you can either allow those jobs to complete normally or you cancel them. canceling an in-flight unit of recovery will result in it being backed out.

Now that there are no Transactional VSAM users of the data set, the data set can be quiesced either using the VARY SMS,SMSVSAM,SPHERE operator command or an equivalent CICS command.

Once the quiesce process completes and the data set has been marked quiesced in the catalog, you can perform whatever operations required the data set to be quiesced.

Once you are done with those operations, the data set should be re-enabled for RLS and Transactional VSAM use. To do this, either issue the VARY SMS,SMSVSAM,SPHERE operator command or an equivalent CICS command.

Quiescing a forward recovery logstream will cause a quiesce of processing for any data sets which use that logstream. Disabling a redo log stream causes all processing which attempts to use that log stream to fail. Transactional VSAM will be unable to commit or back out any units of
recovery which were using the logstream because it will be unable to write the necessary records to the log stream.

Messages IGW473I and IGW474I are issued in response to this command.

QUIESCE: Transactional VSAM should complete processing of any in progress units of recovery using the logstream but not accept any new ones which will require it, except in the case of the log of logs. If the log is a Transactional VSAM system log, it will become quiesced when

all units of recovery which are using Transactional VSAM complete and close any data sets they have open. If the log is a forward recovery log, it will become quiesced when the last data set which uses it and is open for output in Transactional VSAM mode is closed. If the log is a log of logs, it will become quiesced when the last forward recoverable data set which is open for output in Transactional VSAM mode which had written a tie up record to the log of logs is closed. New work may start, but will not write tie up records or file close records to the log of logs.

DISABLE: Transactional VSAM should immediately stop using the logstream. This may prevent completion of commit or back out for units of recovery. Those units of recovery will be shunted, as long as shunting them does not require reading or writing the now disabled log. This command should not be used unless the logstream is damaged or surfacing errors which cannot be corrected without first disabling use of it.

If the log is a Transactional VSAM system log, Transactional VSAM will not allow any further work to be done. All opens and VSAM record management requests are failed. The log will become disabled when all units of recovery which are using Transactional VSAM complete. Transactional VSAM will then retain locks, unregister with RRS and the lock manager, and be unavailable until the log is enabled. No further Transactional VSAM requests can complete until the system log is made available, including commit and back out requests.

If the log is a forward recovery log, any new opens which require the use of the log are failed. The log will become disabled when the last data set which uses it and is open for output in Transactional VSAM mode is closed. If the log is a log of logs, it will become disabled when the last forward recoverable data set which is open for output in Transactional VSAM mode which had written a tie up record to the log of logs is closed. New work may start, but will not write tie up records or file close records to the log of logs.

ENABLE: Transactional VSAM should begin accepting new units of recovery which use the logstream for processing. If Transactional VSAM work was left incomplete when Transactional VSAM processing was stopped, Transactional VSAM will complete that work as part of its restart processing.

VARY SMS,SMSVSAM,SPHERE

This command is used to quiesce or unquiesce a data set for RLS or Transactional VSAM access. The VARY SMS,SMSVSAM,SPHERE(sphere) command was provided in RLS to support unquiescing a data set when CICS was unavailable. This clears the VSAM-quiesced state for the specified sphere. This command is being extended to support quiescing a data set.

A data set might be quiesced to allow it to be accessed in a mode other than RLS or Transactional VSAM, or it might be used to ensure that users do not access the data set while it is being recovered. Before attempting to quiesce a data set, you may want to ensure that all jobs which were accessing the data set using Transactional VSAM are either finished or canceled.

VARY SMS,TRANVSAM(tvsname),PEERRECOVERY

This command is used to start or stop peer recovery processing for a failed instance of Transactional VSAM It applies only to the system on which it is issued. That system will then be responsible for performing all peer recovery processing for the failed Transactional VSAM instance.

ACTIVE: This system should begin peer recovery processing on behalf of the specified failed instance of Transactional VSAM, as long as that instance was not disabling or disabled due to an operator command. It will perform the necessary initialization, and then start tasks to perform any work which was left incomplete by a system failure. Since there could be a large amount of work outstanding, it will start tasks in groups of ten, and then begin more work as those tasks complete. Controlling the amount of work in progress at any given time will allow quiesce of peer recovery processing by varying it INACTIVE in the event that the failed system comes back up.

ACTIVEFORCE: This system should begin peer recovery processing on behalf o f the specified failed instance of Transactional VSAM regardless of the failed instance's status.

INACTIVE: This system should stop processing peer recovery work on behalf of the specified instance of Transactional VSAM. This command does not take affect immediately. Instead, peer recovery processing which is already in progress will be allowed to complete before peer recovery processing terminates.

Forward Recovery Log Errors

In the event that you experience I/O errors on a log stream, the best result you can hope for is to get any existing URs to sync point. This is because if you can reach a commit point you can be certain that the data set is in a consistent state, and it is safe to take a back up of it. Once you have done so, the fact that the log is damaged is no longer relevant because its contents are no longer needed. The new back up of the data set serves the same purpose the forward recovery log records would have served.

For this reason, both CICS and Transactional VSAM allow work that was in progress to continue even after errors have occurred on a log, but they start preventing new work from entering the system which requires the damaged resource. The Transactional VSAM method for doing this is to quiesce the log stream. The CICS method for doing this is to quiesce the data set.

SETSMS Command

The SETSMS command does not support the SYSNAME and TVSNAME parameters. To change these values, update the IGDSMSxx member of SYS1.PARMLIB and issue the SET SMS command.

SETSMS QTIMEOUT

This parameter specifies the quiesce exit timeout value in seconds. This is the amount of time the Transactional VSAM quiesce exit will allow to elapse before it concludes that a quiesce cannot be completed successfully. Changing the value of QTIMEOUT affect only those quiesces which are submitted after the change is made. It has no affect on quiesces which are already in progress. Valid values are 60 to 3600. The default is 300.

SETSMS AKP

This parameter specifies the activity key point trigger value, which is the number of logging operations between the taking of keypoints. Valid values are 200 to 65535. The default is 1000.

SETSMS MAXLOCKS

This parameter specifies two values: the maximum number of unique lock requests that a single unit of recovery may make before warning message IGW859I is issued to the system console and message IGW10074I is issued to the job log, and an increment value. Once the maximum number of unique lock requests is reached, the warning messages will be issued every time the number of unique lock requests over and above the maximum increases by a multiple of the increment. The messages include the job name of the job which is holding the locks. The installation can then determine if the application should be allowed to proceed or if the job should be canceled. If the job is canceled, the UR will be backed out, and the locks will remain held until the back out completes.

Lock requests are considered unique if they lock different records within the base cluster. Repeated requests for the same base cluster records will not result in the count being incremented.

Warning messages IGW859I and IGW10074I are not issued for units of recovery that are in back out. This is because a unit of recovery which is in back out cannot obtain locks on any additional records. In addition, the only actions that can be taken are to allow the job to continue or to cancel the job. Since canceling the job would result in back out, issuing the messages for units of recovery which are already in back out serves no purpose.

Messages IGW859I and IGW10074I are issued until the unit of recovery reaches commit. Once the unit of recovery enters commit, no additional messages will be issued. This is because the unit of recovery can no longer lock any additional records, and the number of locks it holds will not change until the commit is completed, at which point all those locks that can be released will be. Any locks which cannot be released will become retained.

To avoid flooding the system console with messages, messages IGW859I and IGW10074I are issued by an asynchronous timer driven task which wakes up every 10 seconds. This means that the messages will not necessarily reflect the exact values specified for the maximum and the increment, but rather will reflect the values which represent the state of the unit of recovery at the time the task awakens.

Each time it wakes up, it scans the units of recovery that are in progress for any that have exceeded the maximum or which had previously exceeded the maximum and have now exceeded the increment value as well. If it is necessary to issue a new message for a unit of recovery for which a message had previously been issued, the old message will be DOMed from the system console and replaced.

MAXLOCKS takes into account the number of unique lock requests. It does not count the actual number of locks obtained. The number of locks requested will differ from the number of locks held when alternate indexes are used. If an update modifies alternate keys, then a lock is obtained for the base record, for each old alternate key, and for each new alternate key. Therefore, if n alternate keys are modified, a single lock request can result in obtaining (2n+1) locks.

MAXLOCKS is specified as a pair of values in the range of 0 to 999999. The default for both values is 0. A value of 0 indicates that warning messages IGW859I and IGW10074I should not be issued.

If MAXLOCKS is specified but Transactional VSAM is not active, it has not effect. The MAXLOCKS values are sysplex-wide. Changing them on one system changes them on all systems in the sysplex.

SET SMS=xx Command

The SET SMS=xx command causes the IGDSMSxx member of SYS1.PARMLIB to be read. As a result, it can cause changes to any of the parameters which Transactional VSAM is using.

LOG_OF_LOGS

If the log of logs is changed, the new name will be saved, but it will not take effect until the next time Transactional VSAM is restarted. Note that changing it without quiescing the old log of logs before the next Transactional VSAM restart can cause a mismatch of the tie up records written at data set open with file close records.

QTIMEOUT

This is the amount of time the Transactional VSAM quiesce exit will allow to elapse before it concludes that a quiesce cannot be completed successfully. Changing the value of QTIMEOUT affects only affects those quiesces which are submitted after the change is made. It has no affect on quiesces which are already in progress.

TVSNAME

Changing the TVSNAME to SYSNAME mapping takes affect the next time that Transactional VSAM is restarted. Note, however, that changing the mapping should not be done unless all

work under the old mapping completed successfully. Otherwise, it may not be possible for Transactional VSAM to complete recovery of that work.

SYSNAME

See TVSNAME above.

AKP

This parameter specifies the activity key point trigger value, which is the number of logging operations between the taking of keypoints. Valid values are 200 to 65535. The default is 1000.

TV_START_TYPE

This parameter specifies the type of start Transactional VSAM is to perform. It takes affect the next time Transactional VSAM restarts.

Peer Recovery

Peer recovery is very similar to restart processing, in that it completes the processing of any units of recovery which were in progress at the time of the failure. The major difference is that during peer recovery, there are multiple instances of Transactional VSAM running within the SMSVSAM address space: the primary instance, which is working with VSAM RLS to process requests on the system, and an instance which is registered with VSAM RLS and RRS as the failed instance. During peer recovery, Transactional VSAM does the following:

1. It registers with VSAM RLS as the failed instance of Transactional VSAM.
2. It registers with RRS as the failed instance of Transactional VSAM.
3. It invokes RRS to indicate that it is beginning restart processing as the failed instance.
4. It reads the failed instance's undo log to retrieve information about in-progress units of recovery.
5. It invokes RRS to retrieve information about unit of recovery status.
6. It processes the log data and the information returned by RRS to determine whether to commit or back out units of recovery.
7. Indoubt units of recovery are left until RRS determines whether they should be committed or backed out.
8. When all outstanding units of recovery have been processed, it unregisters with RRS and VSAM RLS.

It is necessary for the instance of Transactional VSAM which is performing peer recovery to register as the failed instance in order to gain access to the failed instance's resources. Since only one instance can be registered with a given name at a time, the first system which successfully registers performs the peer recovery.

Note that the registration persists until peer recovery is complete. As a result, should the failed instance restart, it will be unable to initialize because its attempt to register will fail. It may be necessary to restart Transactional VSAM manually once peer recovery is complete.

When Does Peer Recovery Occur?

Due to RRS restrictions concerning where restarts may occur, peer recovery is supported only in cases of system failure. It is not supported when Transactional VSAM fails. This should not be a great restriction, as Transactional VSAM resides within the SMSVSAM address space which is designed to recycle in event of a failure. It also is not supported when the SMSVSAM server fails for much the same reason.

Peer recovery will only occur when both Transactional VSAM and the system on which it was running fail. It will not occur when the Transactional VSAM instance failed, but the system continued to run, because in that case, either Transactional VSAM would automatically restart, or it came down and was meant to stay down.

Peer Recovery and Shared UR Interest

RRS used to require that all resource managers that had shared interest in URs be restarted on the same system. For this reason, Transactional VSAM will use ARM to manage the grouping. If Transactional VSAM tried to manage the grouping, one of three things would happen:

1. Transactional VSAM might, by chance, choose the correct system, in which case all processing should complete successfully.
2. Transactional VSAM might choose the wrong system, but another resource manager had already come up on the correct system, making it impossible for Transactional VSAM to come up.
3. Transactional VSAM might choose the wrong system and come up first, making it impossible for other resource managers to come up on the correct system.

1. A peer recovery Transactional VSAM instance, unlike a normal Transactional VSAM instance, cannot recycle without the SMSVSAM server recycling.

2. A peer recovery Transactional VSAM instance will only be started if there is a primary Transactional VSAM instance of Transactional VSAM running on the system. If Transactional VSAM is not started on the system, then peer recovery will not run.

3. A peer recovery Transactional VSAM instance does not have a quiesce exit of its own. The only Transactional VSAM quiesce exit on the system is the one belonging to the primary Transactional VSAM instance.

4. Peer recovery starts asynchronous tasks to process outstanding URs in parallel. As the tasks complete, additional tasks are started, until all the outstanding URs are processed or an operator command is issued to request termination of peer recovery processing. If such a command is issued, tasks which are already running are allowed to complete, and then peer recovery processing terminates.

5. Peer recovery will be allowed to run if the state of the failed Transactional VSAM instance was quiescing, since peer recovery would only complete the quiesce process. It will not be allowed if the state was quiesced, since in this case there should be no work to do. It also will not be allowed if the state was disabled due to an operator command, since this implies that the installation did not want the Transactional VSAM instance to do any work. If this is not the case and peer recovery should be allowed to run, then use the VARY SMS command and specify the ACTIVEFORCE keyword instead of ACTIVE.

   If the Transactional VSAM instance had been disabling due to an RRS failure, then peer recovery will be allowed to run, since in this case Transactional VSAM would have reinitialized when RRS became available again.

How Do I Start/Stop Peer Recovery

The recommended method of initiating peer recovery processing is to use the automatic restart manager (ARM). To do this, the customer must create an ARM policy which groups instances of Transactional VSAM with other resource managers which may have a shared instance in a unit of recovery. ARM supports this with the administrative policy RESTART_GROUP utility control statement which identifies related elements that are to be restarted as a group if the system on which they are running fails.

Transactional VSAM will register with ARM as an abstract resource (one that is not associated with a job or started task) when it initializes. Transactional VSAM will request that it be restarted only when the system on which it is running fails unexpectedly by specifying an element termination type of SYSTERM and an element bind of CURSYS. Transactional VSAM will provide its restart method by supplying the text of the required VARY SMS command as its start text when it registers with ARM. If Transactional VSAM terminates, it will deregister with ARM.

If the installation is not using ARM or ARM is unavailable, peer recovery can be initiated manually using the VARY SMS command.

Depending on the amount of work left incomplete by a failure, peer recovery may take a significant amount of time. It is therefore possible that the Transactional VSAM instance might attempt to restart while peer recovery is in progress. Since another system is registered with its instance name, its initialization would fail. To prevent this, each Transactional VSAM instance has a resource which it obtains in the exclusive state before attempting to initialize. If it cannot get it, it waits for it to become available. Peer recovery obtains the same resource. It releases it

when it terminates, allowing the Transactional VSAM instance to proceed with initialization.

If peer recovery is running and you need it to stop in order to allow the failed instance of Transactional VSAM to reinitialize, you can stop it at any time by using the VARY SMS command to vary it INACTIVE. Peer recovery will not stop immediately, but instead will complete any tasks it has already started and then terminate without starting any additional tasks. The remaining work will be completed by the failed instance when it reinitializes as part of its restart processing.

Failures During Peer Recovery

Transactional VSAM registers with ARM, requesting that it be restarted only in event of a system failure. Therefore, if the SMSVSAM server in which peer recovery was running fails, ARM cannot automatically restart peer recovery. Instead, Transactional VSAM remembers that it was performing peer recovery by writing that information to the SHCDS. During initialization, it reads this information and restarts any failed
peer recovery work.

A system failure has the effect of terminating both the primary instance of Transactional VSAM running on the system, as well as any peer recovery instances. All instances register with ARM while they are active. Therefore, ARM would recognize the failure and would initiate peer recovery for all involved instances on another system.

If peer recovery is being initiated manually, then a VARY SMS command must be issued on another system. That system will then run peer recovery for the instance specified on the command, as well as any instances for which that instance was performing peer recovery. For example, suppose Transactional VSAM instance IGWTV001 was running peer recovery for instances IGWTV002 and IGWTV003. If the system fails, and peer recovery is initiated for IGWTV001 on another system, that system will perform peer recovery for IGWTV001, IGWTV002, and IGWTV003. However, starting peer recovery for IGWTV002 or IGWTV003 on another system would start peer recovery for the specified instance, and only for the specified instance.

Permit Non-RLS Update

If a data set has retained locks or is in lost locks state, a batch job's non-Transactional VSAM open of the data set will fail. To allow a non-Transactional VSAM batch job to access this data set, the operator can issue the SHCDS PERMITNONRLSUPDATE command. Once the command has been issued, a non-Transactional VSAM batch open will be successful (even if there are retained or lost locks) provided that there are no concurrent RLS or Transactional VSAM opens of the data set.

The SHCDS PERMITNONRLSUPDATE command allows the batch job update or delete records which may be the records for which there are retained locks. Some time after the batch

job has run, Transactional VSAM may be asked to back out the URs for which it holds retained locks. If the back outs are done, Transactional VSAM may invalidate the updates or deletes performed by the batch job. This back out request may come from an in progress UR, Transactional VSAM restarting, the IDCAMS SHCDS RETRY command, or from a communications resource manager issuing a back out request for a UR which had been in indoubt state.

In a Transactional VSAM environment, it will be necessary to quiesce the data set in addition to issuing the PERMITNONRLSUPDATE for it. This is because, unlike CICS, a failure could cause Transactional VSAM to restart. For example, suppose there were five jobs that needed to be run while the PERMITNONRLSUPDATE was active and the data set was not quiesced:

1. The data set has retained locks or is in a lost locks state, but non-RLS and non-Transactional VSAM work needs to be done, so a PERMITNONRLSUPDATE is issued.

2. Job 1 is run and completes successfully.

3. Job 2 is run and completes successfully.

4. A failure occurs which causes Transactional VSAM to restart. As part of restart, Transactional VSAM reads the undo log and encounters the records for the data set which is in PERMITNONRLSUPDATE status. Not knowing that the work is still in progress, Transactional VSAM attempts to perform the back out, invoking the exit as necessary. When it is finished, it clears the PERMITNONRLSUPDATE status. Note that this step should not run until all of the work being done under the PERMITNONRLSUPDATE is complete.

5. Job 3 is run and completes successfully. It is able to open the data set because it no longer has retained locks and it is no longer open for Transactional VSAM access. However, note that the Transactional VSAM restart may have backed out records that this job thought it was supposed to handle.

6. Job 4 is run and completes successfully. Again, Transactional VSAM may have backed out records that this job thought it was supposed to handle.

7. Job 5 is run and completes successfully. Same notes as above.

To prevent this from happening, the data set also needs to be quiesced. This will cause the open that Transactional VSAM would do as part of restart to fail and prevent Transactional VSAM from attempting to perform the back out until the data set is unquiesced.

Exit Environment

The name of the exit must be IGW8PNRU. The module will be loaded by the Transactional VSAM initialization code, and therefore must reside in LINKLIB or LPALIB. If the load fails

(i.e. no exit is found), a message will be issued. If no exit is found, any UR which is requested to be backed out, where a data set involved was accessed via PERMITNONRLSUPDATE, will be shunted.

If the installation needs to fix a code error, or enhance the function of the exit, a restart of Transactional VSAM is required to enable the new exit.

The exit must be loadable from any system which may do peer recovery for another system.

The exit may issue SVC instructions.

Transactional VSAM will establish an ESTAE recovery environment before calling the exit in order to protect the RLS address space from failures in the exit. If the exit fails, the UR will be shunted.

Transactional VSAM Summary

Transactional VSAM provides general VSAM recoverable file sharing. This addresses a long standing requirement for sharing of VSAM files across CICS and batch jobs.