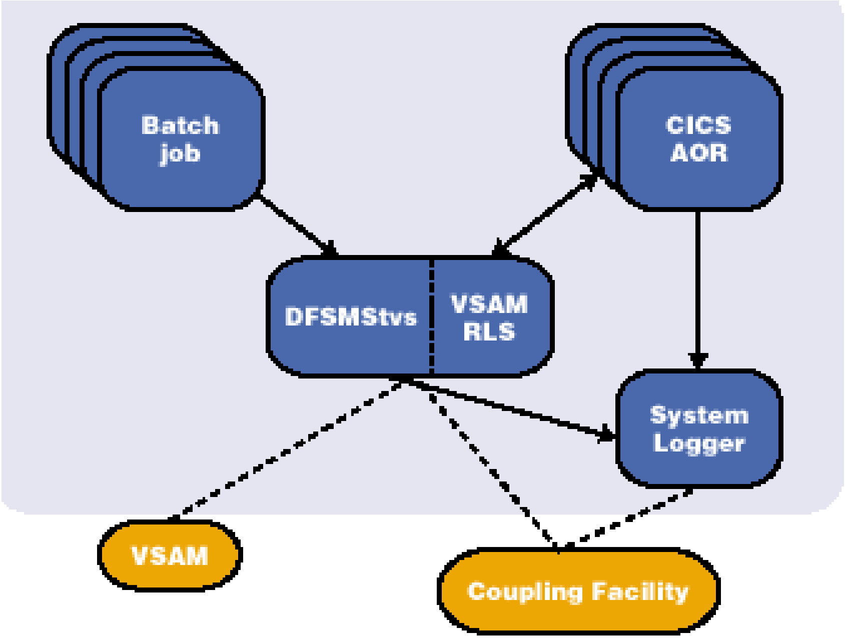
Transactional VSAM: A System Programmer's Perspective



IBM zSeries Technical Conference
Session TSS04
Ruth Ferziger
IBM Design & Development
ruthf@us.ibm.com

Agenda

- What is Transactional VSAM?
- How Transactional VSAM fits into the system
- Setting Up Transactional VSAM
- Peer recovery
- Permit non-RLS Update

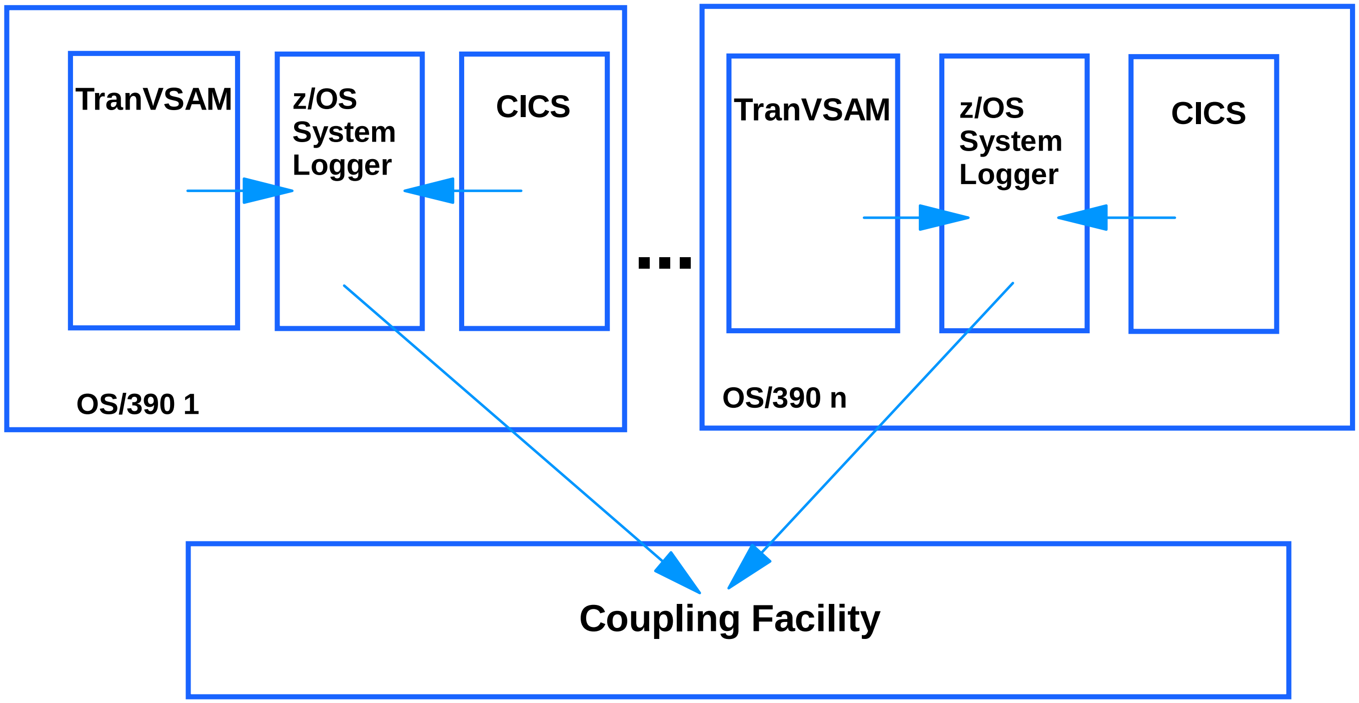
What is Transactional VSAM?

- Objective: Provide transactional recovery within VSAM
- RLS allows batch sharing of recoverable data sets for *read*
  - ► RLS provides locking & buffer coherency
  - ► CICS provides logging & two-phase commit protocols
- Transactional VSAM allows batch sharing of recoverable data sets for *update*
  - ► Logging provided using the MVS System Logger
  - ► Two-phase commit & back out using MVS Recoverable Resource Management Services (RRMS)

Transactional VSAM Overview

- Transactional VSAM uses
  - Recoverable Resource Management Services (RRMS) manages the unit of recovery (UR)
  - System Logger manages the log streams
    - Undo log
    - Shunt log
    - Forward recovery logs and logs of logs
  - VSAM RLS manages locking & buffer coherency
- Allows atomic commit of changes -- all or nothing
- Transactional VSAM does *not* do forward recovery!

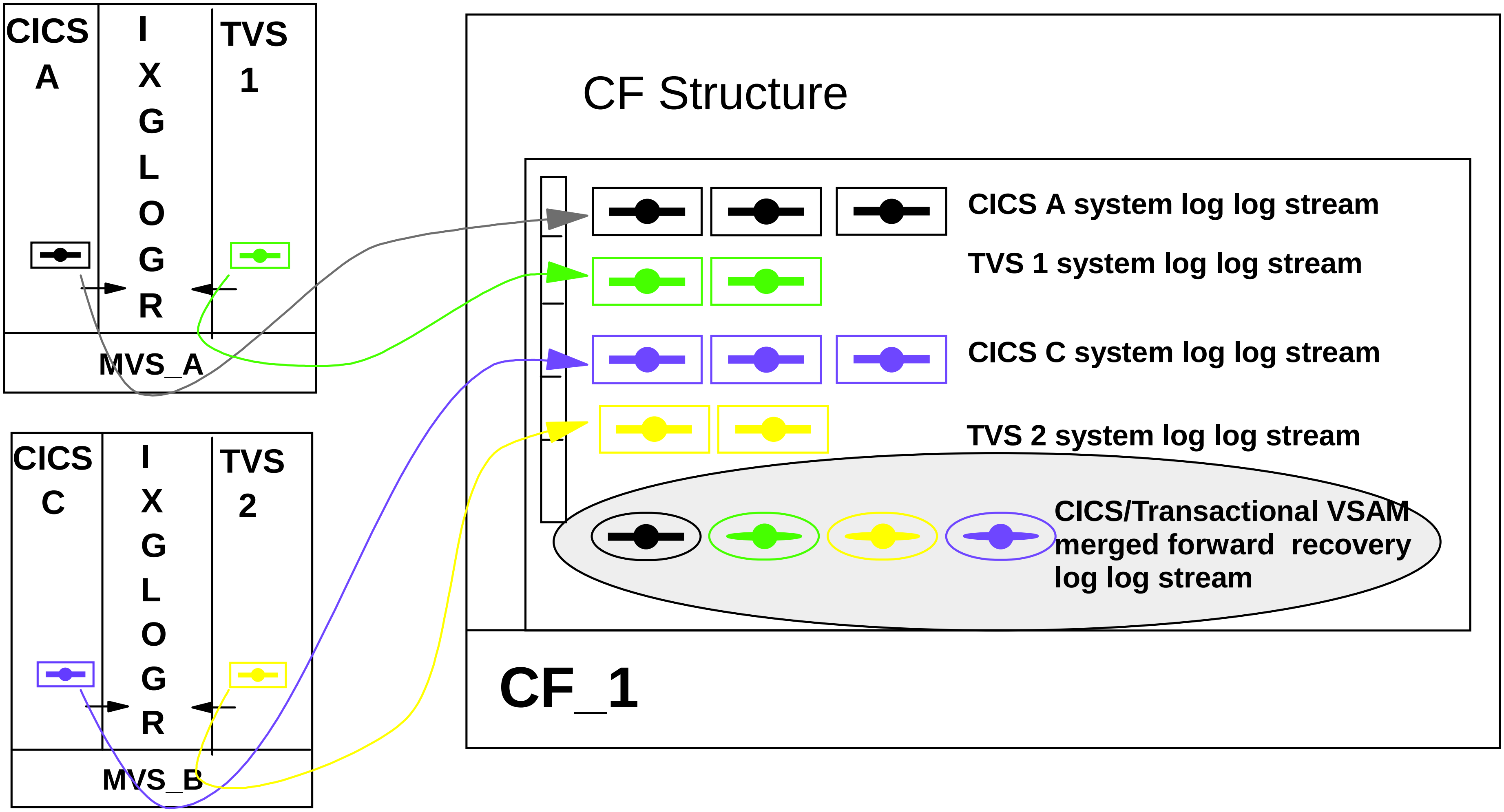
Transactional VSAM Logging



CF List Structures for Logstreams

CF automatic merging of Forward Recovery Logstreams

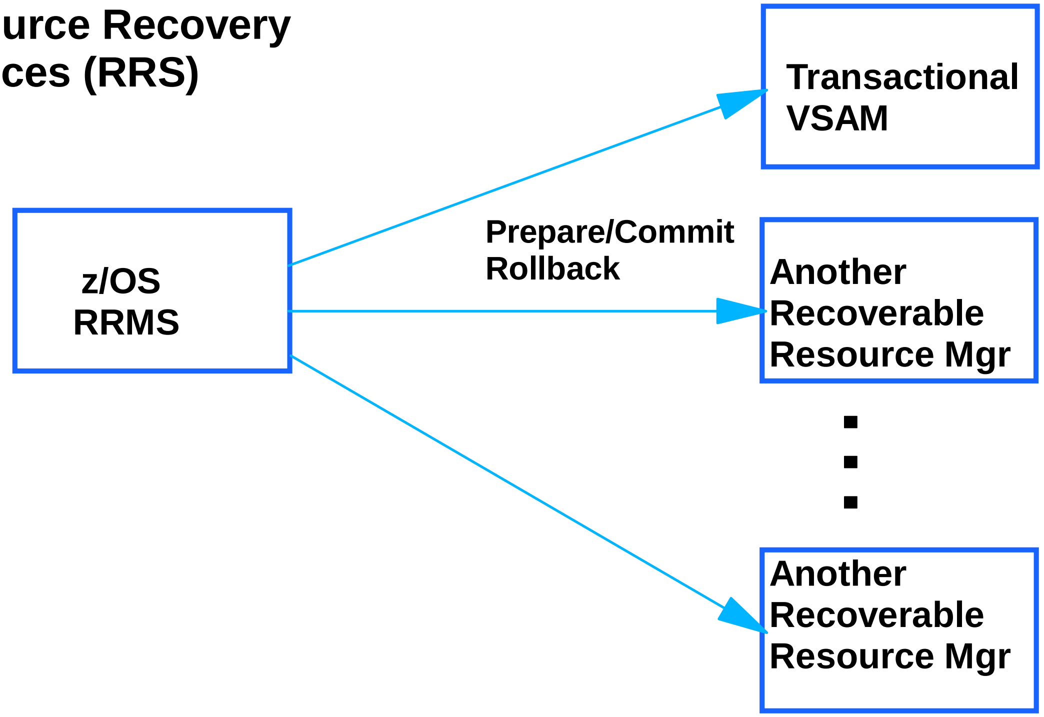
CICS/Transactional VSAM Logstreams


CICS A
IXGLOGR
TVS 1
MVS_A
CICS C
IXGLOGR
TVS 2
MVS_B
CF Structure
CICS A system log log stream
TVS 1 system log log stream
CICS C system log log stream
TVS 2 system log log stream
CICS/Transactional VSAM merged forward recovery log log stream
CF_1

Transactional VSAM & z/OS RRS

z/OS RRMS:

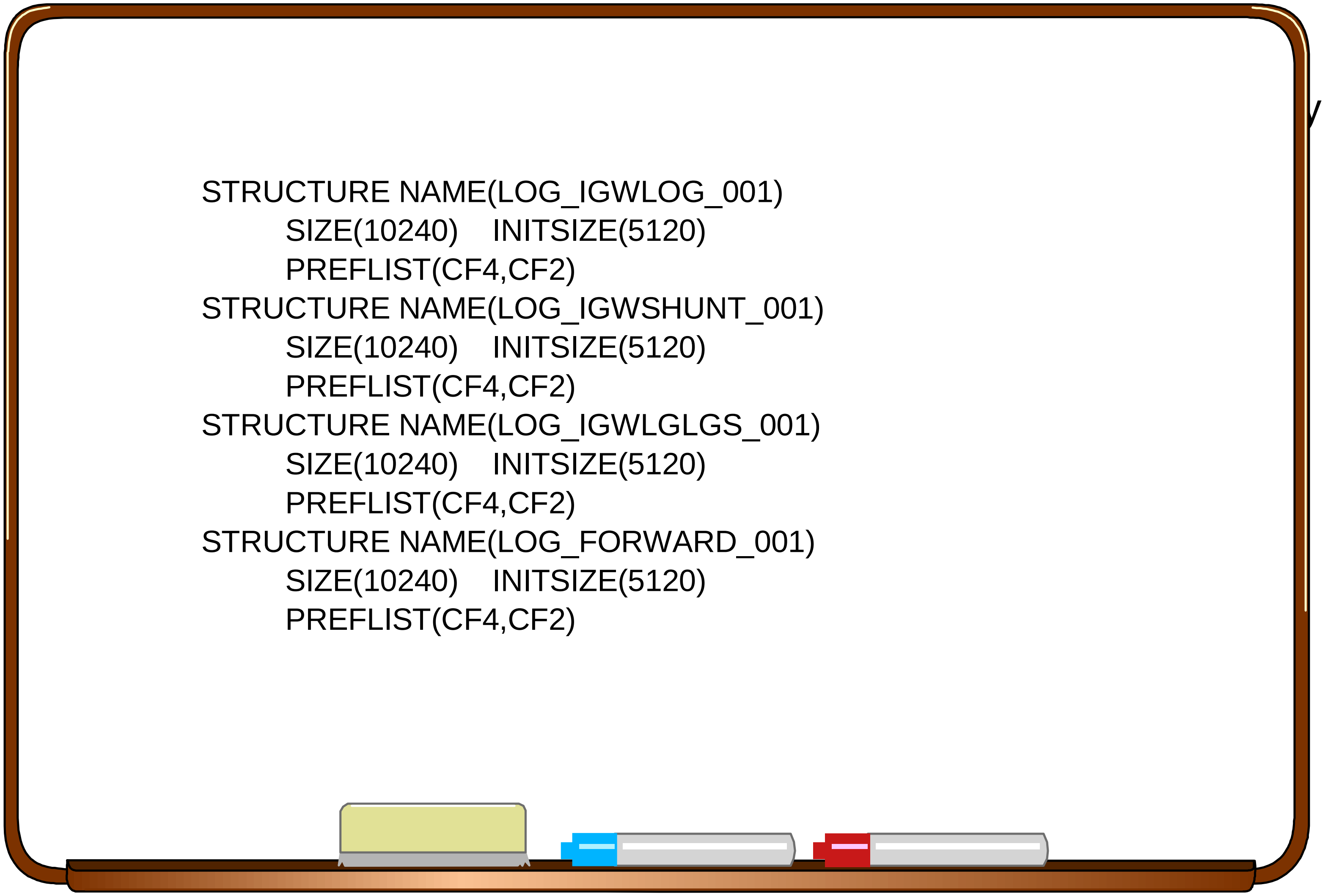
- **Registration Services**
- **Context Services**
- **Resource Recovery Services (RRS)**


z/OS RRMS
Prepare/Commit
Rollback
Transactional VSAM
Another Recoverable Resource Mgr
Another Recoverable Resource Mgr

Defining the Environment

- CFRM Couple Data Set
- Defining Log Streams
- RACF Security
- SMS Constructs

CFRM Couple Data Set



```
STRUCTURE NAME(LOG_IGWLOG_001)
        SIZE(10240)    INITSIZE(5120)
        PREFLIST(CF4,CF2)
STRUCTURE NAME(LOG_IGWSHUNT_001)
        SIZE(10240)    INITSIZE(5120)
        PREFLIST(CF4,CF2)
STRUCTURE NAME(LOG_IGWLGLGS_001)
        SIZE(10240)    INITSIZE(5120)
        PREFLIST(CF4,CF2)
STRUCTURE NAME(LOG_FORWARD_001)
        SIZE(10240)    INITSIZE(5120)
        PREFLIST(CF4,CF2)
```

Define Log Structures

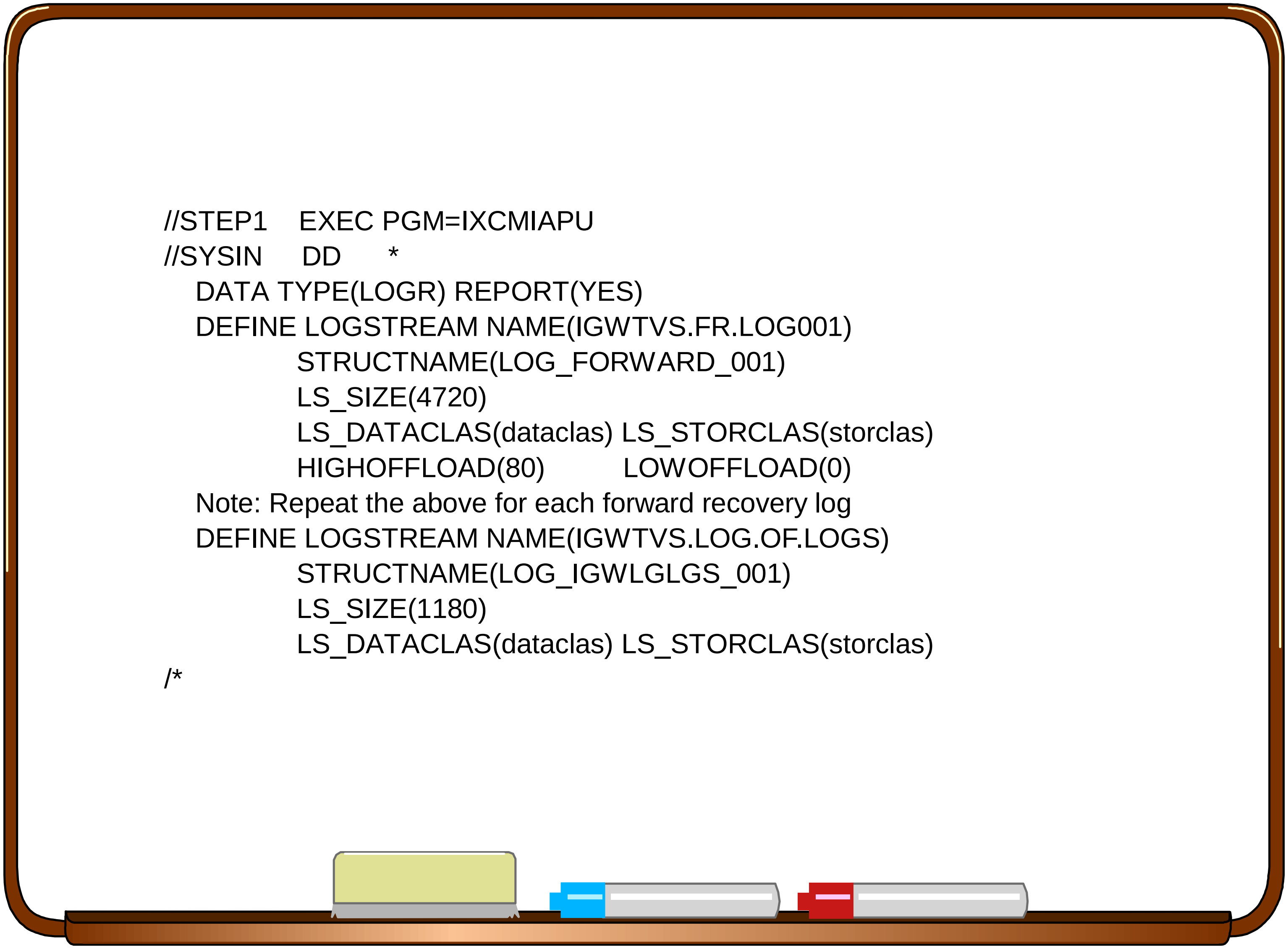
```
//STEP1    EXEC PGM=IXCMIAPU
//SYSIN     DD      *
   DATA TYPE(LOGR) REPORT(YES)
   DEFINE STRUCTURE NAME(LOG_IGWLOG_001)
                LOGSNUM(10) MAXBUFSIZE(64000)
                AVGBUFSIZE(4096)
   DEFINE STRUCTURE NAME(LOG_IGWSHUNT_001)
                LOGSNUM(10) MAXBUFSIZE(64000)
                AVGBUFSIZE(4096)
   DEFINE STRUCTURE NAME(LOG_IGWLGLGS_001)
                LOGSNUM(10) MAXBUFSIZE(64000)
                AVGBUFSIZE(4096)
   DEFINE STRUCTURE NAME(LOG_FORWARD_001)
                LOGSNUM(20) MAXBUFSIZE(64000)
                AVGBUFSIZE(4096)
/*
```

Define Logstreams

```
//STEP1    EXEC PGM=IXCMIAPU
//SYSIN     DD      *
   DATA TYPE(LOGR) REPORT(YES)
   DEFINE LOGSTREAM NAME(IGWTV001.IGWLOG.SYSLOG)
                STRUCTURENAME(LOG_IGWLOG_001)
                LS_SIZE(1180)
                LS_DATACLAS(dataclas)  LS_STORCLAS(storclas)
                STG_DUPLEX(YES)        DUPLEXMODE(COND)
                HIGHOFFLOAD(95)        LOWOFFLOAD(15)
                DIAG(YES)

   DEFINE LOGSTREAM NAME(IGWTV001.IGWSHUNT.SHUNTLOG)
                STRUCTURE(LOG_IGWSHUNT_001)
                LS_SIZE(100)
                LS_DATACLAS(dataclas) LS_STORCLAS(storclas)
                HIGHOFFLOAD(95)       LOWOFFLOAD(15)
                DIAG(YES)
/*
```

Define Log Streams....



```
//STEP1    EXEC PGM=IXCMIAPU
//SYSIN     DD      *
    DATA TYPE(LOGR) REPORT(YES)
    DEFINE LOGSTREAM NAME(IGWTVS.FR.LOG001)
                STRUCTNAME(LOG_FORWARD_001)
                LS_SIZE(4720)
                LS_DATACLAS(dataclas) LS_STORCLAS(storclas)
                HIGHOFFLOAD(80)          LOWOFFLOAD(0)
    Note: Repeat the above for each forward recovery log
    DEFINE LOGSTREAM NAME(IGWTVS.LOG.OF.LOGS)
                STRUCTNAME(LOG_IGWLGLGS_001)
                LS_SIZE(1180)
                LS_DATACLAS(dataclas) LS_STORCLAS(storclas)
/*
```

RACF Definitions for Transactional VSAM Logstreams

- Associate the SMSVSAM address space with a RACF defined userid in the started procedures table
- Protecting the log streams from general access:
  - ► RDEFINE LOGSTRM IGWTV001.** UACC(NONE)
- Granting access to the log streams:
  - ► PERMIT IGWTV001.** CLASS(LOGSTRM) ACCESS(UPDATE) ID(smsvsam_userid)
  - ► PERMIT IGWTV001.** CLASS(LOGSTRM) ACCESS(READ) ID(authorized_browsers)
  - ► PERMIT IGWTV001.** CLASS(LOGSTRM) ACCESS(UPDATE) ID(archive_userid)

RACF Definitions for Forward Recovery Logstreams

- RDEFINE LOGSTRM FORWARD.RECOVERY.** UACC(NONE)
- RDEFINE LOGSTRM FR.LOG.** UACC(NONE)
- RDEFINE LOGSTRM LOG.OF.LOGS UACC(NONE)
- PERMIT FORWARD.RECOVERY.** CLASS(LOGSTRM) ACCESS(UPDATE) ID(smsvsam_userid)
- PERMIT FR.LOG.** CLASS(LOGSTRM) ACCESS(UPDATE) ID(smsvsam_userid)
- PERMIT LOG.OF.LOGS CLASS(LOGSTRM) ACCESS(UPDATE) ID(smsvsam_userid)

RACF Definitions for Peer Recovery

- Permit all Transactional VSAM instances access to each other's system logs
- Do the following for each instance:
  - ► PERMIT IGWTV001.** CLASS(LOGSTRM) ACCESS(UPDATE) ID(smsvsam_userid)
  - ► PERMIT IGWTV002.** CLASS(LOGSTRM) ACCESS(UPDATE) ID(smsvsam_userid)
  - ► PERMIT IGWTV003.** CLASS(LOGSTRM) ACCESS(UPDATE) ID(smsvsam_userid)
  - ► Etc.

SMS Constructs

- Explicit definition of DATACLAS and STORCLAS
- May also want to specify MGMTCLAS
  - Log archiving
  - Log data back up
  - Log migration
- Log stream and log stream staging data sets are single extent VSAM linear data sets (shareoptions '3,3')
  - Requires an active SMS address space**.**
- All data sets must be accessible to all members of the sysplex that required for peer recovery.

SYS1.PARMLIB Changes

SMS ACDS(acds)
INTERVAL(nnn|15)
REVERIFY(YES|NO)
SYSTEMS(8|32)
SIZE(nnnnnK|M)
JOBNAME(jobname|*)
SELECT(event,event....)
DSNTYPE(LIBRARY|PDS)
RLSINIT(NO|YES)
SMF_TIME(NO|YES)
BMFTIME(nnn|3600)
DEADLOCK_DETECTION(iii|15,kkk|4)
SYSNAME(sys1,sys2....)
TV_START_TYPE(WARM|COLD,WARM|COLD...)
LOG_OF_LOGS(logstream)
MAXLOCKS(max|0,incr|0)

COMMDS(commds)
DINTERVAL(nnn|150)
ACSDEFAULTS(YES|NO)
TRACE(OFF|ON)
TYPE(ALL|ERROR)
ASID(asid|*)
DESELECT(event,event....)

RLS_MAX_POOL_SIZE(nnn|100)
CF_TIME(nnn|3600)
CACHETIME(nnn|3600)
RLSTMOUT(nnn|0)
TVSNAME(nnn1,nnn2....)
AKP(nnn|1000,nnn|1000)
QTIMEOUT(nnn|300)

SYS1.PARMLIB Rules

- If any Transactional VSAM parameter is specified but there is no TVS name, it is a syntax error.
- If TVSNAME is specified without SYSNAME, only one value may be specified.
- If more than one TVSNAME is specified but SYSNAME is not, it is a syntax error.
- If Transactional VSAM parameters are specified on a system not specified by SYSNAME, they will be ignored.
- All parameters that specify multiple values must specify the same number of values.
- Syntax errors due to a mismatch in the number of values specified will be flagged on all systems.

About the MAXLOCKS Parameter

- Specifies two values: maximum unique locks value and increment value
- Values control issuance of warning messages IGW859I and IGW10074I
- Lock requests are unique if they lock different records within the base cluster
- Warning messages are not issued for unit of recovery in sync point (commit or back out)

SYS1.PARMLIB Examples

Example	Valid?
TVSNAME(1)	Yes. Applies only to one system. All other parameters are defaulted.
TVSNAME(1) AKP(500)	Yes. Applies only to one system. All parameters other than AKP are defaulted.
TVSNAME(1) AKP(500,200)	No. The TVSNAME parameter specifies only one value while AKP specifies two.
TVSNAME(1) SYSNAME(SYS1) RLSTMOUT(15)	Yes. Still applies only to one system. All parameters except RLSTMOUT are defaulted.
TVSNAME(1,2) SYSNAME(SYS1,SYS2) QTIMEOUT(500)	Yes. Applies to two systems. Parameter except QTIMEOUT are defaulted.
QTIMEOUT(500) AKP(500,200)	No. The TVSNAME parameter is required when Transactional VSAM parameters are specified.
TVSNAME(1,2) AKP(500,200) QTIMEOUT(500)	No. The SYSNAME parameter is required when more than one TVSNAME value is specified.

SYS1.PARMLIB Examples....

EXAMPLE	VALID?

TVSNAME(1, 2, 3)
SYSNAME(SYS1,SYS2,SYS3)
TV_START_TYPE(WARM,COLD)
	No. TVSNAME and SYSNAME specify three values while TV_START_TYPE specifies only two.

TVSNAME(1, 2, 3)
SYSNAME(SYS1,SYS2,SYS3)
TV_START_TYPE(WARM,COLD,WARME)
	No. The right number of values is specified on each parameter, but WARME is not a valid value for TV_START_TYPE.

TVSNAME(1, 2, 3)
SYSNAME(SYS1,SYS2,SYS3)
TV_START_TYPE(WARM,COLD,WARM)
LOG_OF_LOGS(LOG.OF.LOGS)
	Yes.

TVSNAME(1, 2, 3)
SYSNAME(SYS1,SYS2,SYS3)
TV_START_TYPE(COLD,,WARM)
AKP(,700,900)
	Yes. The second TV_START_TYPE value and the first AKP value are defaulted.

TVSNAME(1, 2, 3)
SYSNAME(SYS1,SYS2,SYS3)
TV_START_TYPE(WARM,COLD,)
	Yes. The third TV_START_TYPE value is defaulted.

MAXLOCKS Examples

SPECIFICATION	VALID?
MAXLOCKS(0,0)	Yes. Warning messages are never issued.
MAXLOCKS(,)	Yes. Equivalent to specifying MAXLOCKS(0,0).
MAXLOCKS(5000,0)	Yes. Warning messages are issued after a UR has made its 5000th lock request.
MAXLOCKS(2300,)	Yes. Warning messages are issued after a UR has made it 2300th lock request.
MAXLOCKS(0,2000)	No. An increment value may not be specified without a maximum value.
MAXLOCKS(,2000)	No. An increment value may not be specified without a maximum value.

MAXLOCKS Examples....

SPECIFICATION	VALID?
MAXLOCKS(4500,1000)	Yes. Warning messages are issued after the UR has made its 4500th lock request and again after every subsequent 1000.
MAXLOCKS(1000,2300)	Yes. Warning messages are issued after the UR has made its 1000th lock request and again after every subsequent 2300.

There is no rule that says the increment value must be less than the maximum value.

DISPLAY SMS Command

```
DISPLAY SMS     ,TRANVSAM                  {,ALL}
                ,JOB(jobname)
                ,URID(urid|ALL)
                ,SHUNTED,                  {SPHERE(sphere)|
                                            URID(urid|ALL)}
                ,LOG(logstream| ALL)
                ,DSNAME(dsn)
                ,OPTIONS
```

VARY SMS Command

```
VARY SMS   ,TRANVSAM(tvsname|ALL),     {QUIESCE | Q}
                                       {DISABLE | D}
                                       {ENABLE | E}
           ,LOG(logstream),            {QUIESCE | Q}
                                       {DISABLE | D}
                                       {ENABLE | E}
           SMSVSAM,SPHERE(sphere),     {QUIESCE | Q}
                                       {ENABLE | E}
           ,TRANVSAM(tvsname),         {ACTIVE |
              PEERRECOVERY,             ACTIVEFORCE |
                                        INACTIVE}
```

Quiescing a Data Set

- Display the jobs using the data set:
  - DISPLAY SMS,DSNAME
  - IDCAMS SHCDS LISTDS(dsname),JOBS
- Allow the jobs to complete or cancel them
- Quiesce the data set
  - VARY SMS,SMSVSAM,SPHERE
  - Equivalent CICS command
- Perform the operations which required the data set to be quiesced.
- Enable the data set for RLS/Transactional VSAM use:
  - VARY SMS,SMSVSAM,SPHERE
  - Equivalent CICS command

Forward Recovery Log Errors

- Transactional VSAM quiesces the log
- CICS quiesces the data set
- The two mechanisms are *not* compatible
  - If any jobs have the data set open for Transactional VSAM access, Transactional VSAM fails the quiesce

SETSMS Command

```
SETSMS     AKP(nnn | 1000)
           QTIMEOUT(nnn | 300)
           MAXLOCKS(max | 0, incr | 0)
```

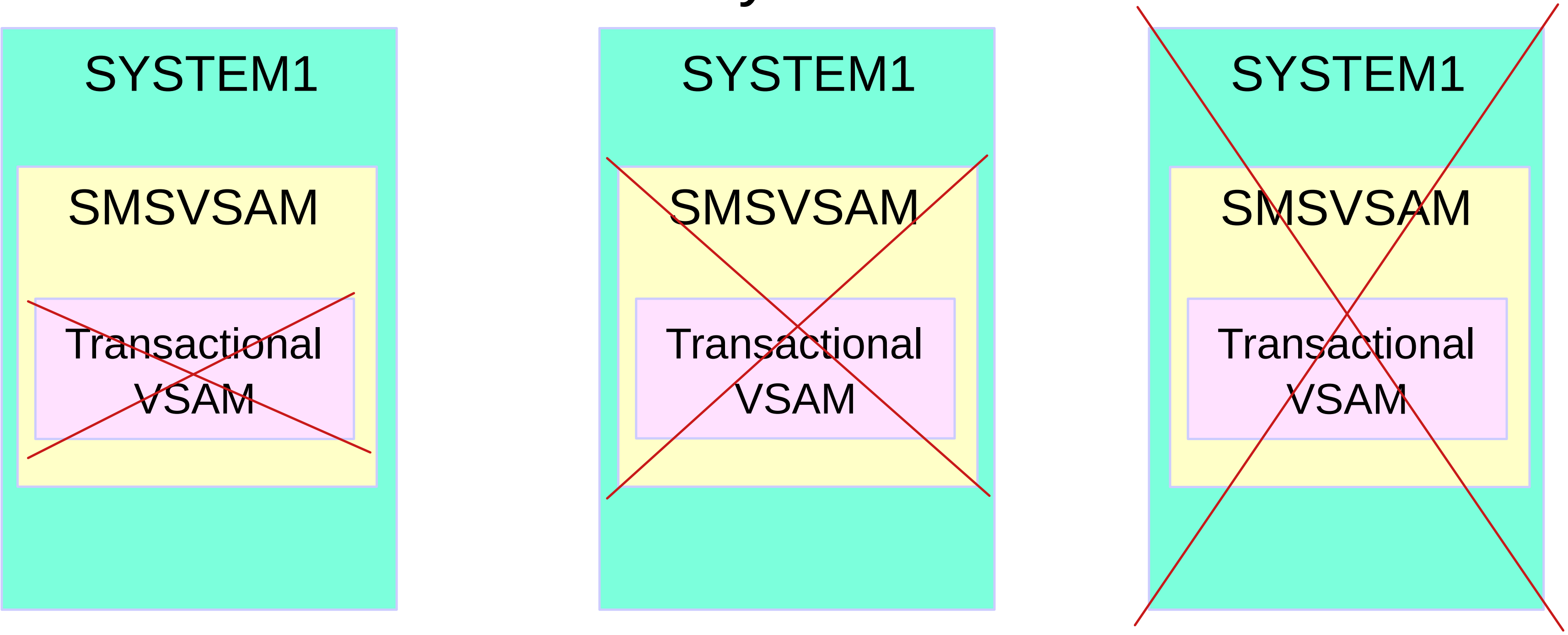
SET SMS=xx Command

- Causes the IGDSMSxx member of SYS1.PARMLIB to be read
- Can be used to change:
  - Log of logs
  - QTIMEOUT
  - TVSNAME/SYSNAME mapping
  - AKP
  - TV_START_TYPE
- Most changes take affect the next time Transactional VSAM restarts

Peer Recovery

- Allows another instance of Transactional VSAM to recover for a failed instance
- Cleans up any work that was left in an incomplete state
- Clears retained locks that resulted from the failure
- Does *not* accept any **new** work!
- Since peer recovery instance must register as failed instance, failed instance cannot restart while peer recovery is in progress

When Does Peer Recovery Occur?

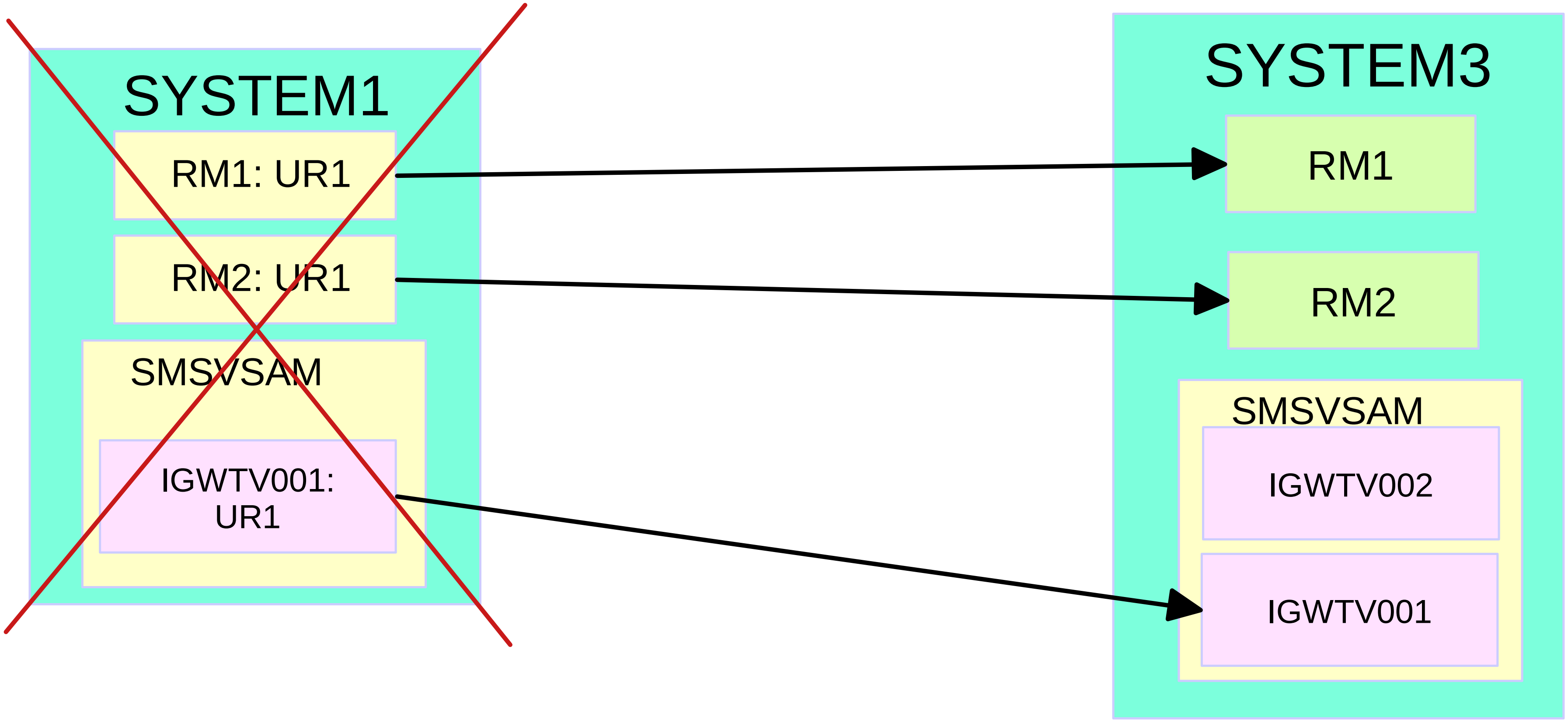


No

No

Yes

Peer Recovery and Shared UR Interest....



How Do I Start/Stop Peer Recovery?

- Recommended start method: Automatic Restart Manager policy
- Alternative start method:
  - ► VARY SMS,TRANVSAM(nnn),PEERRECOVERY,ACTIVE
  - ► Optionally specify the ACTIVEFORCE keyword
- To stop peer recovery:
  - ► VARY SMS,TRANVSAM(nnn),PEERRECOVERY,INACTIVE

Failures During Peer Recovery

- SMSVSAM server failure
  - Peer recovery restarts when SMSVSAM restarts
- System failure
  - ARM restarts all failed instances (primary and peer recovery) on other systems
  - Peer recovery can also be initiated via operator command

Permit Non-RLS Update

- Data set has retained locks or is in lost locks state
- Non-Transactional VSAM, non-RLS batch update job needs to be run
- Without PermitNonRLSUpdate, batch open would fail
- Specified on IDCAMS SHCDS command
- Data set should also be quiesced
- Must supply a batch override exit: IGW8PNRU

Exit Environment

Interrupts	Enabled
State	Problem program
Key	8
ASC Mode	P=H=S, SMSVSAM address space
AMODE, RMODE	No restrictions
Locks	None held
Reentrancy	Must be reentrant
Serialization requirements	None

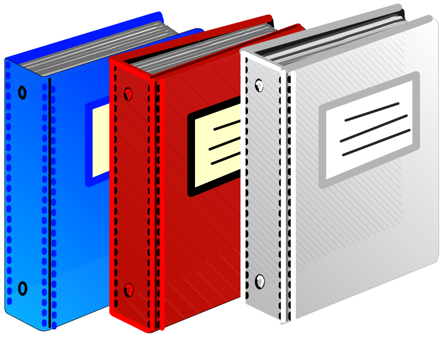
Registers

Registers on entry	Reg 0	Not applicable
	Reg 1	Points to undo log record (IGWUNLR) in key 8 storage
	Reg 2	Points to an area in key 8 to be used as an autodata area
	Reg 3	Contains the length of the autodata area
	Reg 4-13	Not applicable
	Reg 14	Return address
	Reg 15	Address of entry point for exit
Registers on exit	Reg 0-14	Not applicable
	Reg 15	Return code: - 0 - do not back out this record - 4 - back out this record

Transactional VSAM Summary

- Transactional VSAM is a general purpose VSAM recoverable file manager for z/OS
- Transactional VSAM addresses a long-standing CICS batch window requirement
  - Read/write sharing across CICS and batch jobs
  - Read/write sharing across multiple batch jobs
  - Major steps towards 24x7 CICS on-line availability
- Setting up Transactional VSAM involves:
  - Changes to SYS1.PARMLIB
  - Defining log structures and log streams
  - Setting up RACF authorization

Appendix A.



References

References

- ***z/OS MVS Setting Up a Sysplex*** **- SA22-7625**
- ***z/OS MVS Programming: Resource Recovery*** **- SA22-7616**
- ***z/OS MVS Diagnosis: Tools and Service Aids*** **- GA22-7589**
- ***z/OS MVS Programming: MVS Assembler Services Guide - SA22-7605***
- ***z/OS MVS Programming: Authorized Assembler Services Reference, Volume 2 - SA22-7610***
- ***OS/390 Parallel Sysplex Configuration Cook Book, Volumes 1-3*** **- SG24-2075, SG24-2076, SG24-2077**
- ***CICS Recovery and Restart Guide,*** **SC33-1698**

Notices & Disclaimers

Copyright © 2003 by International Business Machines Corporation.

No part of this document may be reproduced or transmitted in any form without written permission from IBM Corporation.

Product information and data has been reviewed for accuracy as of the date of initial publication. Product information and data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Consult your local IBM representative or IBM Business Partner for information about the product and services available in your area.

Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectually property rights, may be used instead. It is the user's responsibility to evaluate and verify the operation of any non-IBM product, program or service.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR INFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. IBM is not responsible for the performance or interoperability of any non-IBM products discussed herein.

Notices & Disclaimers

The performance data contained herein was obtained in a controlled, isolated environment. Actual results that may be obtained in other operating environments may vary significantly. While IBM has reviewed each item for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere.

The responsibility for use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's or user's ability to evaluate and integrate them into their operating environment. Customers or users attempting to adapt these techniques to their own environments do so at their own risk. IN NO EVENT SHALL IBM BE LIABLE FOR ANY DAMAGE ARISING FROM THE USE OF THIS INFORMATION, INCLUDING BUT NOT LIMITED TO, LOSS OF DATA, BUSINESS INTERRUPTION, LOSS OF PROFIT OR LOSS OF OPPORTUNITY.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not necessarily tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Trademarks

DFSMSdfp, DFSMStvs, DFSMSdss, DFSMShsm, DFSMSrmm, IBM, IMS, MVS, MVS/DFP, MVS/ESA, MVS/SP, MVS/XA, OS/390, SANergy, and SP are trademarks of International Business Machines Corporation in the United States, other countries, or both.

AIX, CICS, DB2, DFSMS/MVS, Parallel Sysplex, OS/390, S/390, Seascape, and z/OS are registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Domino, Lotus, Lotus Notes, Notes, and SmartSuite are trademarks or registered trademarks of Lotus Development Corporation. Tivoli, TME, Tivoli Enterprise are trademarks of Tivoli Systems Inc. in the United States and/or other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both. UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

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

Appendix B.

Operator Command Examples

DISPLAY SMS,TRANVSAM Example

If Transactional VSAM is not active:

```
IGW800I    DFSMS VSAMRLS REQUEST TO DISPLAY TRANVSAM INFORMATION
IS REJECTED:  TRANVSAM IS NOT ACTIVE ON THIS SYSTEM
```

If Transactional VSAM is active:

```
IGW800I    11.52.18 DISPLAY SMS,TRANVSAM

DISPLAY SMS,TRANVSAM - SERVER STATUS
 System   TvsName  State   Rrs    #Urs     Start     AKP    QtimeOut
 -------- -------- ------ ----- -------- --------- -------- --------
 SYSTEM1  IGWTV001 ACTIVE REG          9 WARM/WARM     1000      400

DISPLAY SMS,TRANVSAM - LOGSTREAM STATUS
 LogStreamName              State      Type       Connect Status
 -------------------------- ---------- ---------- --------------
 IGWTV001.IGWLOG.SYSLOG     Enabled    UnDoLog    Connected
 IGWTV001.IGWSHUNT.SHUNTLOG Enabled    ShuntLog   Connected
 IGWTVS1.LOG.OF.LOGS        Enabled    LogOfLogs  Connected
```

DISPLAY SMS,JOB Example

If the job is not using Transactional VSAM:

```
IGW801I DFSMS REQUEST TO DISPLAY TRANVSAM USAGE OF
JOB: jobname WAS REJECTED. JOB IS NOT KNOWN
TO ANY TRANVSAM INSTANCE IN THE SYSPLEX.
```

If the job is using Transactional VSAM:

```
IGW801I    12.42.25 DISPLAY SMS,JOB

TRANVSAM Job Status On System: SYSTEM1
JobName  StepName Urid                             Ur Status  # Locks
-------- -------- -------------------------------- ---------- --------
jobname  stepname urid                             Status            0
```

DISPLAY SMS,URID Example

If the URID is not active:

```
IGW802I   DFSMS REQUEST TO DISPLAY ACTIVE TRANVSAM  UR(s)
WAS REJECTED,  SPECIFIED URID(s) ARE NOT ACTIVE
ON ANY TRANVSAM INSTANCE IN THE SYSPLEX.
```

If the URID is active:

```
IGW802I 12.49.24 DISPLAY SMS,Ur

TRANVSAM UrId Status For System: SYSTEM1
Urid                             Ur Status  JobName  StepName # Locks
-------------------------------- ---------- -------- -------- --------
Urid                             Status     Jobname  Stepname        0
```

DISPLAY SMS,SHUNTED Example

If there is no shunted work:

```
IGW803I    09.50.47 DISPLAY SMS,UR (Summary Data)

SysName   # Urid(s) SysName   # Urid(s) SysName   # Urid(s)
--------  --------  --------  --------  --------  --------
SYSTEM1          0  --------         0  --------         0
```

If there is shunted work:

```
IGW803I    09.50.47 DISPLAY SMS,UR (Summary Data)

SysName   # Urid(s) SysName   # Urid(s) SysName   # Urid(s)
--------  --------  --------  --------  --------  --------
SYSTEM1   urid      -------          0  --------         0
```

DISPLAY SMS,LOG Example

If Transactional VSAM is not currently using the log:

```
IGW804I DFSMS REQUEST TO DISPLAY
TRANVSAM LOG STREAM:  logstream  WAS REJECTED.
LOG STREAM NOT KNOWN TO DFSMS.
```

If Transactional VSAM is using the log:

```
IGW804I 13.06.52 DISPLAY SMS,LOG

DISPLAY SMS,LOG      - LOG STREAM STATUS
  Name: IGWTV001.IGWLOG.SYSLOG     State: Enabled      Type: UnDo
  System   TvsName  JobName  Urid of Oldest Log Block
  -------- -------- -------- --------------------------------
  SYSTEM1  IGWTV001 jobname  urid                           *

DISPLAY SMS,LOG      - LOG STREAM USAGE
  LogStreamName: IGWTV001.IGWLOG.SYSLOG
  System   TvsName  JobName  JobName  JobName  JobName  JobName
  -------- -------- -------- -------- -------- -------- --------
  SYSTEM1  IGWTV001 jobname  -------- -------- -------- --------
*OLDEST URID ACROSS ALL SYSTEMS IN THE SYSPLEX
```

DISPLAY SMS,DSNAME Example

If the data set is not open in Transactional VSAM mode:

```
IGW805I DFSMS REQUEST TO DISPLAY TRANVSAM USAGE OF
DATASET: dsname WAS REJECTED.
DATASET NOT KNOWN TO TRANVSAM.
```

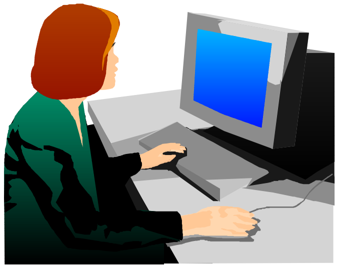
If the data set is open in Transactional VSAM mode:

```
IGW805I 13.15.04 DISPLAY SMS,DSNAME

DATASET:  dsname
IS CURRENTLY IN USE BY THE FOLLOWING JOBS:

System Name:  SYSTEM1  TvsName:  IGWTV001
JobNames:     jobname  --------  -------- -------- -------- --------
```

Appendix C.



IDCAMS SHCDS Commands

RLS Commands

SHCDS	LISTDS(base-cluster)
	LISTSUBSYS(subsystem|ALL)
	LISTSUBSYSDS(subsystem|ALL)
	LISTRECOVERY(base-cluster)
	LISTALL
	FRSETRR(base-cluster)
	FRRESETRR(base-cluster)
	FRUNBIND(base-cluster)
	FRBIND(base-cluster)
	FRDELETEUNBOUNDLOCKS(base-cluster)

RLS Commands....

SHCDS	PERMITNONRLSUPDATE(base-cluster)
	DENYNONRLSUPDATE(base-cluster)
	REMOVESUBSYS(subsystem)
	CFREPAIR(INFILE(ddname))
	CFREPAIR(INDATASET(dsname))
	CFRESET(INFILE(ddname))
	CFRESET(INDATASET(dsname))
	CFRESETDS(base-cluster)

Transactional VSAM Commands

SHCDS	LISTDS(base-cluster) {JOBS}
	LISTSHUNTED SPHERE(base-cluster)
	LISTSHUNTED URID(urid|ALL)
	RETRY SPHERE(base-cluster)
	RETRY URID(urid)
	PURGE SPHERE(base-cluster)
	PURGE URID(urid)

SHCDS LISTDS Example

```
  SHCDS LISTDS(SYSPLEX.KSDS.RETAINED.CLUS1) JOBS
----- LISTING FROM SHCDS ----- IDCSH02
--------------------------------------------------------------------
DATA SET NAME----SYSPLEX.KSDS.RETAINED.CLUS1
  CACHE STRUCTURE----CACHE01
  RETAINED LOCKS----------NO   NON-RLS UPDATE PERMITTED---------NO
  LOST LOCKS--------------NO   PERMIT FIRST TIME----------------NO
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES

                           SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  RETLK05A     ONLINE--ACTIVE    YES         NO       NO

  JOB NAMES:
      TRANV001    TRANV002    TRANV003    TRANV004    TRANV005
      TRANJOB1    TRANJOB2    TRANJOB3    TRANJOB4    TRANJOB5

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

SHCDS LISTSHUNTED Example

```
    SHCDS LISTSHUNTED SPHERE(SYSPLEX.KSDS.CLUSTER.NAME)
--------------------------------------------------------------------
CLUSTER NAME----SYSPLEX.KSDS.CLUSTER.NAME

URID                DISPOSITION    JOB NAME    STEP NAME    CAUSE
----------------    -----------    --------    ---------    -----
ABCDEFGH00000001    BACKOUT        TRANJOB1    TRANSTP3     B-FAILED
XYZ@#$0000000000    BACKOUT        TRANJOB2    STPTRAN1     IO-ERROR
0101BF$$22222222    COMMIT         TRANV001    TRANSTP1     C-FAILED
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```