#SHAREorg

System Logger Update

Nicholas R. Jones
IBM z/OS System Logger

Thursday, March 15, 2012: 9:30- 10:30AM
Session Number 10848

This presentation is for educational purposes only.
It is not a deliverable to customers.

Trademarks

- See url http://www.ibm.com/legal/copytrade.shtml for a list of trademarks.

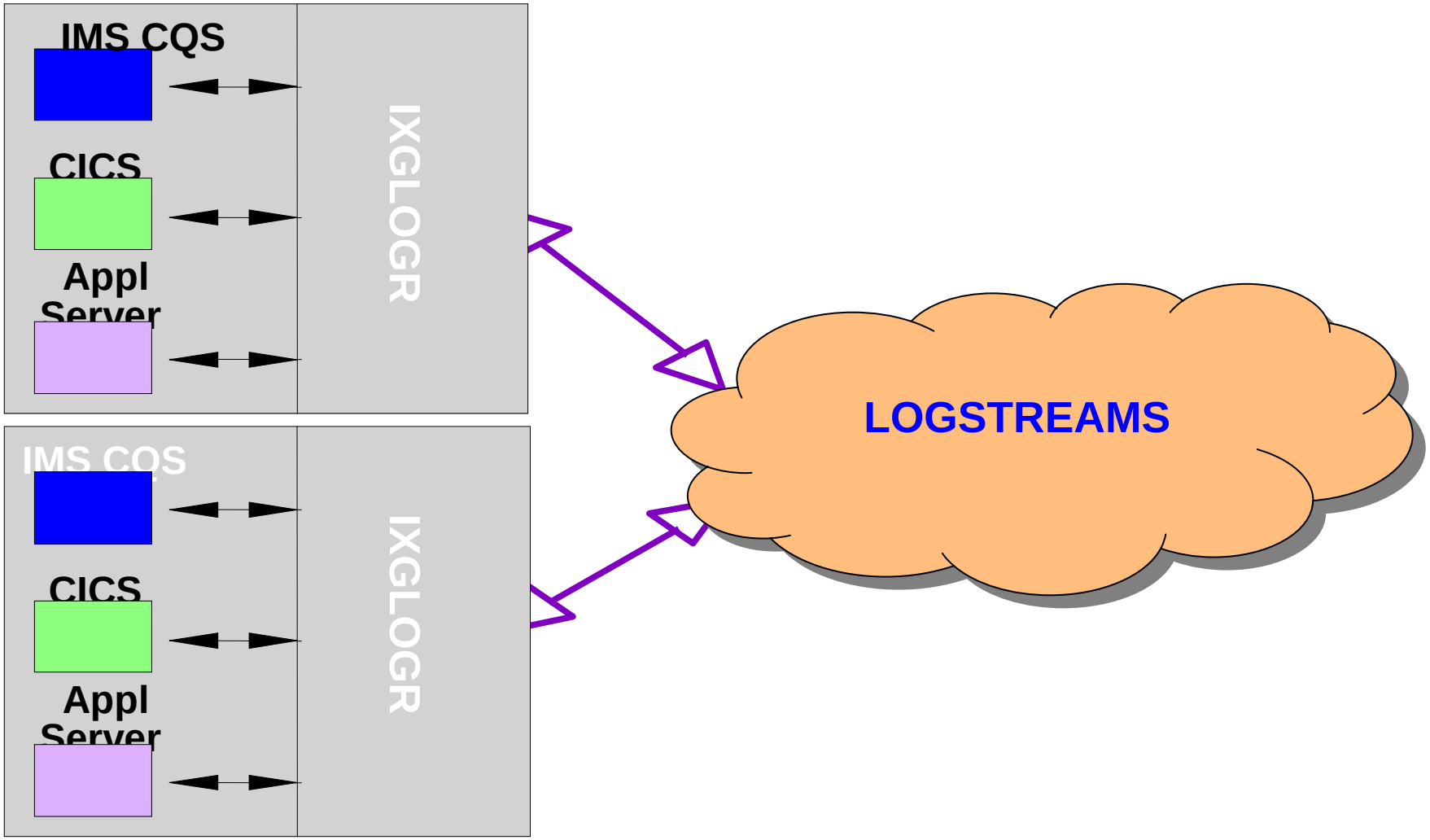
Table of Contents

- Logger Review
- Component History
- Recent Enhancements
  - Problem determination
    - What went wrong
    - Tell me sooner
  - Recover from failures
  - Prevent failures
  - Data set management
  - Tasking improvements (get out of our own way)

Logger Review

What is logger and what does it do?

Logger Review: 50,000 Foot View



Logger Review: What is it?

- An MVS BCP component
- Provides
  - Logging access method services for applications to log data
    - into a log stream sysplex resource
    - from multiple systems (or one system)
- Advantages
  - Merging log data real-time
  - Providing continuous availability for log data
  - Integrity of data and resources
  - Simplifying log management

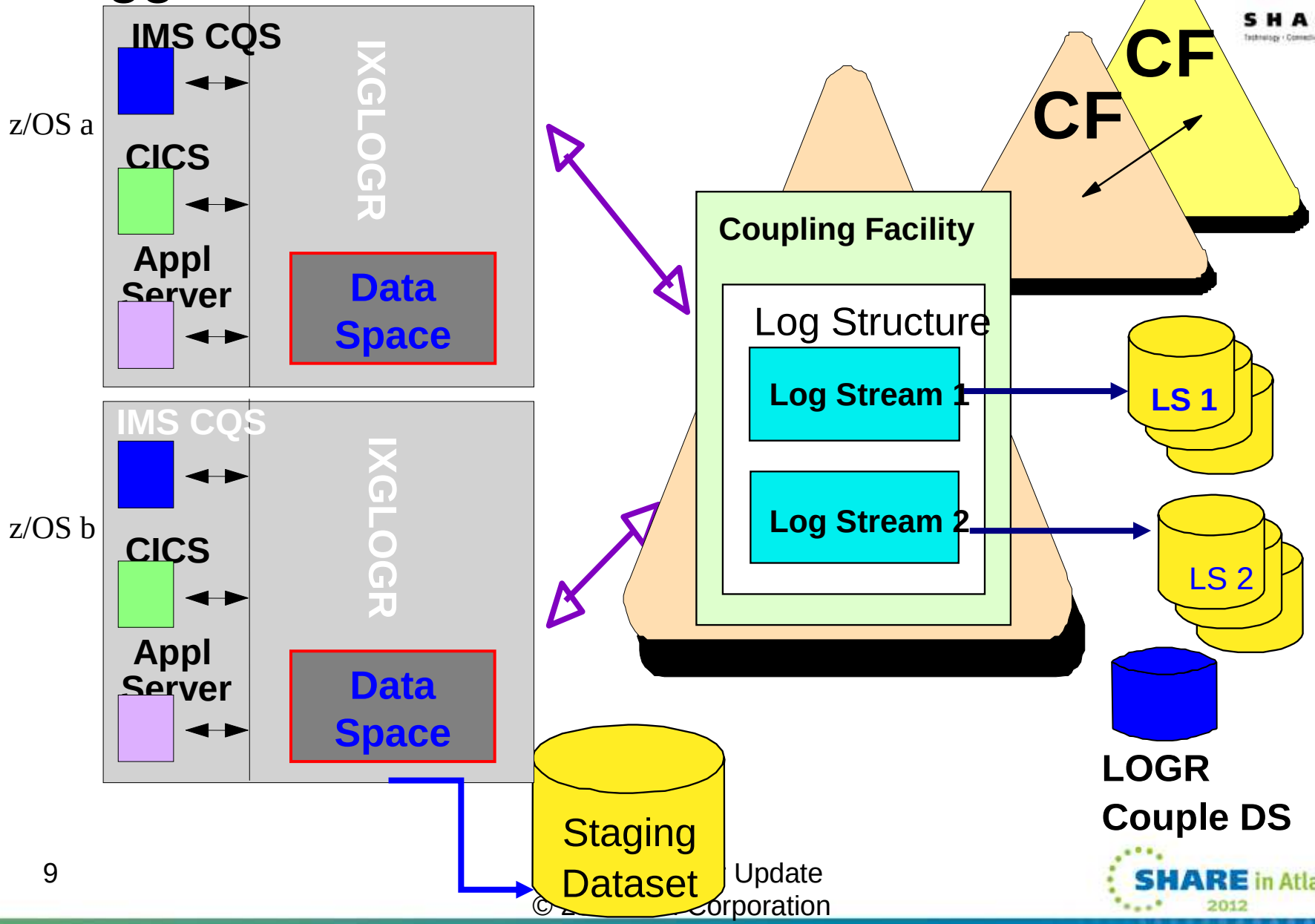
Logger Review: Types of log streams

- Two types of log streams
  - Coupling Facility structure
    - multi-system scope
    - data is first written to a CF structure and later offloaded to an offload dataset
  - DASD-only
    - single system scope only
    - data is first written to local storage buffers and later offloaded to an offload dataset

Logger Review: What Does it do?

- Manages z/OS resources for log stream exploiters at system and/or sysplex scope
- Provides multiple configuration combinations
- Handles system and/or CF failure recovery as well as CF structure rebuilds
- Must be in a z/OS sysplex environment !
  - Can be single or multi-system sysplex
  - Can use Coupling Facility (CF) resources (parallel sysplex)
  - Will use other sysplex resources (non-CF) (base sysplex)

Logger Review: Closer Look



Logger Review: Resources

- 1 system address space per z/OS image - IXGLOGR
- 1 LOGR CDS per sysplex - XCF services
- CF structures – CFRM - XES services
- VSAM linear datasets - Allocation/DFSMS
  - Staging datasets (temporary for recovery purposes)
  - Offload datasets (permanent)
  - Can be migrated (e.g. DFHSM)

Logger Review: Log streams

- A 'log stream' is a collection of data
- Data in a log stream may reside in multiple storage media (CF structures, system data space local buffers, DASD datasets, tape)
- User view of the log stream is:
  - A set of records in time sequence order, merged into a single stream
  - Independent of physical media residence
- The installation or application determines placement of log stream data
- Log block - A logger term used to represent a single piece of application data that was successfully written to the log stream and has been assigned a log block identifier (8 byte token value)

Logger Review: Log stream storage

- Interim (Primary) Storage
  - CF structures, data space local buffers
  - Data can be accessed quickly without incurring DASD I/O
- Secondary (DASD Dataset) storage
  - Offload datasets
  - Data is hardened for longer term access
  - Can be migrated to tertiary storage

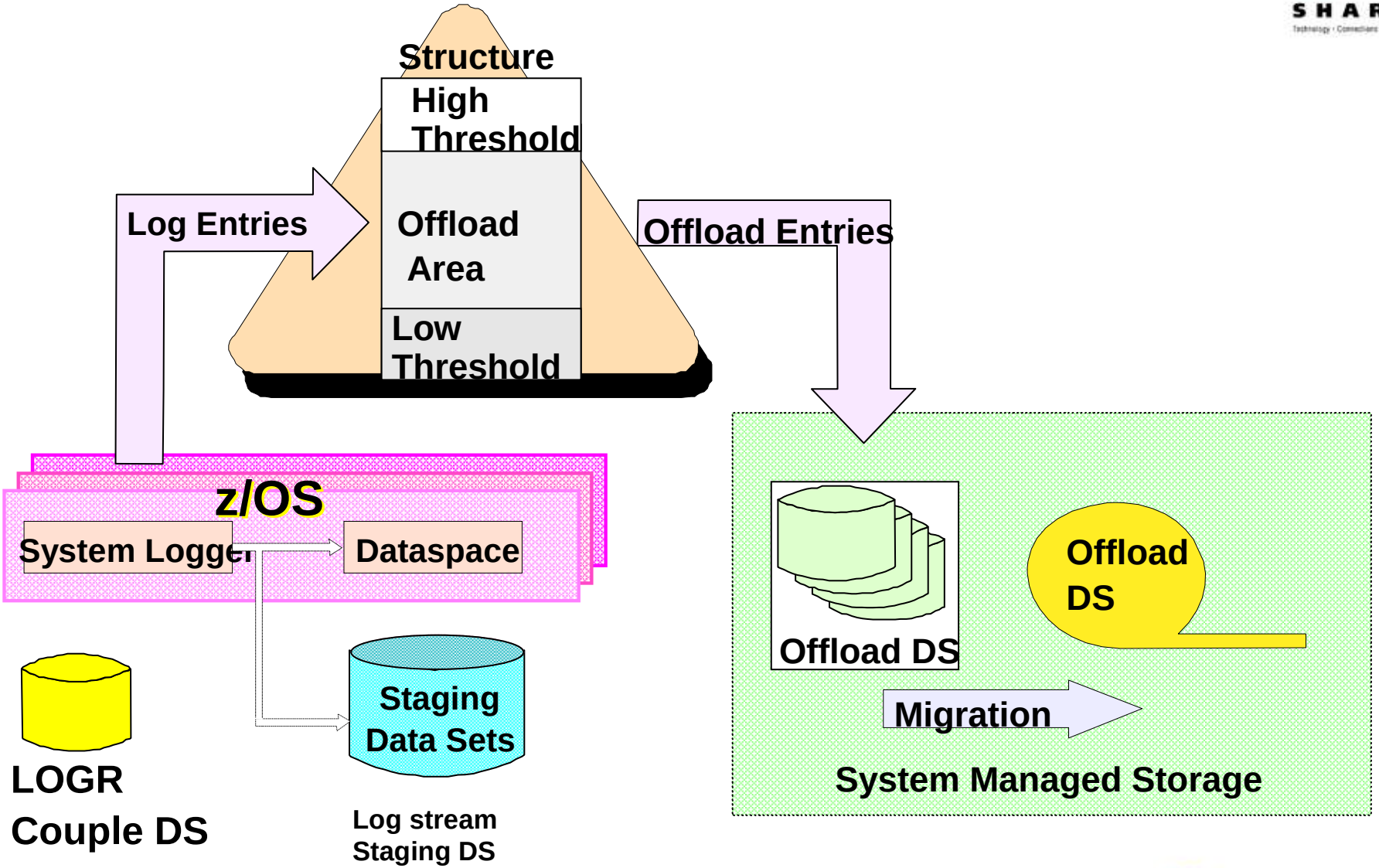
Logger Review: Duplexing

- Each time an application requests to write to a logstream, System Logger ensures there is a duplicate copy of the data to protect against data loss
- Methods of duplexing data for a CF structure-based logstream.
  - local buffers
  - duplicate CF structure
  - staging dataset
- Methods of duplexing data for a DASD-only logstream
  - staging datasets
- Note:
  - Once the data has been offloaded from interim storage, it is no longer duplexed.
  - Offloaded data in DASD is considered *hardened*, and only one copy of the log data exists (notwithstanding mirrored DASD)

Logger Review: Offload

- Process of moving data from interim storage to DASD offload datasets
- What causes an offload?
  - The log stream's CF structure or staging dataset begins to fill and reaches or exceeds the defined HIGHOFFLOAD threshold
  - IXGOFFLD operation
  - System, CF Failure, Structure Rebuild Processing, Recovery processing
- One Logger instance performs the offload and calculates the amount of data to be offloaded
- The data is retrieved from the CF or local buffers and written to DASD offload datasets
- Other Logger instances are notified to reclaim space in the staging dataset or local buffers if needed

Logger Review: Offload



Logger Review: Application Developer Concerns

- Interfaces:
  - IXGCONN – Connect/Disconnect to Log Stream
  - IXGWRITE – Write Log Data to a Log Stream
  - IXGBRWSE – Read Log Data
  - IXGDELET – Deleting Log Data
  - IXGINVNT – Managing the LOGR Inventory Couple Data Set
  - IXGQUERY – Query a Log Stream Info
  - IXGOFFLD – Initiate Offload to DASD Log Data Sets
  - IXGIMPRT – Import Log Blocks
  - IXGUPDAT – Update Log Stream Control Information
- Concerns
  - Synchronous VS Asynchronous processing
  - Temporary Error Conditions
  - Browsing – Gaps and Deleted blocks
  - Reason Code 804
  - Answer Area
  - ENF 48 listening (Authorized callers)
  - Completion Exit (Authorized callers)
  - SSI calls

Logger Review: System Programmer Concerns

- Setup
  - SAF authorization
  - CFRM and LOGR CDS setup
  - Log stream definition management (IXCMIAPU utility)
  - Plan DASD Space (DFSMS classes)
    - How much log data
    - How long to keep the data
  - Logger Trace setup
- Management
  - Tuning
    - SMF type 88 Records
- Commands
  - D LOGGER
  - SETLOGR
- Messages
  - Offload monitoring messagesIXG310, IXG311, IXG312
  - Task Monitoring messages IXG271, IXG272
- Problems
  - Doc Collection
  - Recovery
  - Cleanup

Logger Review: Who uses it?

- Within z/OS BCP:
  - Logrec (OBR)
  - Operlog (Consoles syslog)
  - z/OS Health Checker
  - SMF (System Management Facility)
  - eWLM (Enterprise Work Load Manager)
  - RRS (MVS Resource Recovery Services)
  - APPC (APPC/MVS Protected Conversations)
- Outside of z/OS BCP:
  - CTS (CICS Transaction Server)
  - CICS VSAM RLS (record level sharing)
  - IMS/ESA CQS (common queue services)
  - TVS (transactional VSAM)
  - Web Sphere Application Server Services
  - Web Sphere II Classic Event Publisher for VSAM
  - Problem Determination Work Bench (CEA– common event adapter)
  - ISVs and client home grown…

Logger Component History

… back in the day

Component History: 1995 MVS SP 5.2.0

- Logger was once the new component on the street
- In 1995, MVS SP 5.2.0, logger was introduced as a new component.
- Initial version used XES services to write to CF structure based logstreams, duplexing to either staging data sets or local buffers.
- Dasdonly log streams weren't available
- Very few externals, no display, no report, -- very difficult to see what was going wrong

Component History: 1997

- OS390 V1R3 (1997)
  - New LOGR CDS format (HBB6603 format level)
  - Removal of 168 limit of offload data sets per logstream (DSEXTENT)
  - Tail management / archive support (RETPD keyword)
  - Logger management (alter) of structure entry/element ratio (AVGBUFSIZE keyword)
- OS390 V2R4 (1997)
  - DASDONLY log streams

Component History: 1998-1999

- OS390 V2R6 (1998)
  - SMF88 records for data flow
  - IXCMIAPU LIST DETAIL(YES) enhancements
  - Offload performance
    - 24k CI sizes for offload data sets
  - Local buffer performance improvement
- OS390 V2R8 (1999)
  - Dumping enhancements
  - D XCF,C,TYPE=LOGR output

Component History: 2000-2004

- OS390 V2R10 (2000)
  - Browse Readcursor Multiblock
  - Latching improvements
- z/OS V1R2 (2001)
  - CF System-Managed Duplexing
  - New Logger CDS Level (HBB7705)
- z/OS V1R3 (2002)
  - Dynamic log stream parameter updates
- z/OS V1R4 (2002)
  - Offload Monitoring
- z/OS V1R5 (2004)
  - Task Monitoring
  - Data Set Deletion monitoring
  - IXGOFLDS, IXGDELAB, IXGDELLS procs
  - Share options (3,3) checking
- z/OS V1R6 (2004)
  - 64 Bit caller support
  - 64 bit IXGBRWSE and IXGWRITE buffer support

Recent Enhancements

While MVS System logger has not been presented at Share in several years, many improvements have taken place

Recent Enhancements

- Areas of focus:
  - Problem determination
    - What went wrong
    - Tell me sooner
  - Recover from failures
  - Prevent failures
  - Data set management
  - Tasking improvements (get out of our own way)

Problem Determination

What went wrong?

Problem Determination

- What went wrong?
  - IXGRPT2 (v1R13)
  - IXCMIAPU report enhancements (V1R10)
  - Log Stream Latch Identity (V1R11)

- Tell me sooner!
  - IXGCNF Parmlib, offload monitoring interval updates (V1R13)
  - Logger Health Checks (OA15593, OA22255)

IXGRPT2 (v1R13)

- New SYS1.SAMPLIB member to format SMF type 88 subtype 1 and 11 records using ICETOOL
- Alternate to old IXGRPT1
- Allows for ease of customization
- Paired with new SYS1.SAMPLIB member IXGRPTMP
  - Maps the SMF type 88 records in a DFSORT / ICETOOL readable format
- Follow prologue for suggested modifications

IXGRPT2 (v1R13) sample output REPORT01

```
LOGGER 88-1 ACTIVITY REPORT        05/12/11        10:01:54        - 1 -

LOGSTREAM  CQS.FF.LOGSTRM
```

TME	DTE	SYN	LWI	LIB	LAB
00:00:00	2011/05/12	JA0	314277	64	8488
00:00:00	2011/05/12	JC0	6	64	8488
00:00:00	2011/05/12	JB0	354048	64	12344
00:00:00	2011/05/12	J90	397986	64	11904
00:30:00	2011/05/12	JA0	319994	64	8488
00:30:00	2011/05/12	JC0	0	64	8488
00:30:00	2011/05/12	JB0	348883	64	12344
00:30:00	2011/05/12	J90	393898	64	11904
01:00:00	2011/05/12	JA0	277879	64	8488
01:00:00	2011/05/12	JC0	0	64	8488
01:00:00	2011/05/12	J90	405855	64	11904
01:00:00	2011/05/12	JB0	330528	64	12344
01:30:00	2011/05/12	JA0	286414	64	8488
01:30:00	2011/05/12	JC0	0	64	8488
01:30:00	2011/05/12	JB0	335133	64	12344
01:30:00	2011/05/12	J90	399723	64	11904
02:00:00	2011/05/12	JA0	277099	64	8488
02:00:00	2011/05/12	JC0	0	64	8488
02:00:00	2011/05/12	J90	396046	64	11904
02:00:00	2011/05/12	JB0	345174	64	12344
02:30:00	2011/05/12	JA0	279610	64	8488
02:30:00	2011/05/12	JC0	0	64	8488
02:30:00	2011/05/12	JB0	335746	64	12344
02:30:00	2011/05/12	J90	402855	64	11904
03:00:00	2011/05/12	JA0	285699	64	8488
03:00:00	2011/05/12	JC0	0	64	8488
03:00:00	2011/05/12	J90	409007	64	11904

…

IXGRPT2 (v1R13) sample output REPORT11

```
1LOGGER 88-11 DETAIL SECTION        05/12/11        10:01:59        - 1 -

 STRUCTURE NAME  LG2_DFHJ04_B
       DTE        TME   SYN              ATK        ATK                    AWB            AO           ACB
----------   --------   --------   ----------   --------   --------------------   -----------   -----------
2011/05/12   00:00:00   JA0        1900/01/01   00:00:00                      0             0           512
2011/05/12   03:30:16   JA0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   03:30:16   JE0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   03:30:16   JC0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   03:30:16   JF0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   03:30:16   J80        1900/01/01   00:00:00                      0             0          1280
2011/05/12   03:30:16   J90        1900/01/01   00:00:00                      0             0          1280
2011/05/12   03:30:16   Z0         1900/01/01   00:00:00                      0             0          1280
2011/05/12   03:30:16   JB0        2011/05/12   07:30:01                      0             0          1280
2011/05/12   04:00:00   JA0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:00:00   JE0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:00:00   J80        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:00:00   JC0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:00:00   JF0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:00:00   Z0         1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:00:00   JB0        2011/05/12   07:30:01                      0             0          1280
2011/05/12   04:00:00   J90        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:30:00   JE0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:30:00   JF0        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:30:00   J80        1900/01/01   00:00:00                      0             0          1280
2011/05/12   04:30:00   JA0        1900/01/01   00:00:00                      0             0          1280
…
```

IXGRPT2 (v1R13) (IXGRPT1 field reference)

```
BYT WRITTN  BYT WRITTN  BYT WRITTN                                          AVERAGE
                                          BY USERS   TO INTERIM    TO DASD    #WRITES    ---# WRITES COMPLETED------   BUFFER
--LOGSTREAM NAME------------ STRUCTURE NAME--  IXGWRITES   STORAGE                  INVOKED     TYPE1     TYPE2     TYPE3    SIZE
                                         BYT DELETD  # DELETES   BYT DELETD  # DELETS    ---------------EVENT----------------
                                         INTERIM ST   W/O DASD   INTERIM ST     W/      OFF- DASD STRC NTRY STG  STG   RE-
                                         W/O DASD     WRITE      W/DASD        WRITE    LOAD SHFT FULL FULL THLD FULL  BLD


  SMF88LSN                   SMF88STN         SMF88LWB    SMF88SWB    SMF88LDB    SMF88LWI     SMF88SC1  SMF88SC2  SMF88SC3  AVG_BUF
                                              SMF88SIB    SMF88SII    SMF88SAB    SMF88SAI    EO   EDS  ESF  EFS  ETT  ETF   ERI
```

IXCMIAPU report enhancements (V1R10)

- Problems faced
  - Clients requested more info in the list log stream report
  - When one request fails all subsequent requests aren’t run
  - When specifying DASDONLY(YES) you can't specify the default duplexing options
- New fields displayed
- Continue Keyword
- Default duplexing options with DASDONLY(YES) allowed
  - STG_DUPLEX(YES)
  - DUPLEXMODE(UNCOND)
  - LOGGERDUPLEX(UNCOND)

IXCMIAPU report enhancements (V1R10) New fields displayed

```
LOGSTREAM NAME(USER01.STREAM.NAME) STRUCTNAME()
   …

LOG STREAM ATTRIBUTES:

   User Data: 0000000000000000000000000000000000000000000000000000000000000000
              0000000000000000000000000000000000000000000000000000000000000000

   Time Defined: 02/25/02 17:32:22 (GMT)

…

LOG STREAM DATA SET INFO: DATA SET NAMES IN USE: IXGLOGR.USER01.STREAM.NAME.<SEQ#>

    Ext.  <SEQ#>   Lowest Blockid / Highest GMT /     Highest Local /   Status
                   Highest Blockid  Highest RBA       System Name
    ----- -------- ---------------- ----------------- ----------------- ---------
   *00001 A0000166 000000000F000001 02/25/02 18:48:31 02/25/02 13:48:31 DELETE PENDING
                   000000000F012B6C 00013BA0          SYSTEM_1
          A0000167 000000000F013BA1 02/25/02 18:48:32 02/25/02 13:48:31 DELETED
                   000000000F0266EB 00013BA0          SYSTEM_5
   .00002 A0000168 000000000F027741 02/25/02 18:48:32 02/25/02 13:48:31 CURRENT
                   000000000F02E45A 00007D5C          SYSTEM_1
```

IXCMIAPU report enhancements (V1R10) CONTINUE keyword added

- New request verb CONTINUE
  - A new request CONTINUE will be allowed to run subsequent commands even after a previous failure. For example if we ran this job while stream1 still had a connection:
  -
```
//LSDEL JOB
//STEP1 EXEC PGM=IXGMIAPU
//SYSIN DD * DATA TYPE(LOGR) REPORT(YES)
   CONTINUE
   DELETE LOGSTREAM NAME(stream1)
   DELETE LOGSTREAM NAME(stream2)
   DELETE LOGSTREAM NAME(stream3)
   DELETE LOGSTREAM NAME(stream4)
   DELETE LOGSTREAM NAME(stream5)
   DELETE LOGSTREAM NAME(stream6)
```
  - Log stream stream1 would not be deleted, but the utility would continue to process the delete commands for stream2 through stream6.

IXCMIAPU report enhancements (V1R10) CONTINUE keyword output

```
ADMINISTRATIVE DATA UTILITY:  INPUT                        DATA TYPE = LOGR
         LINE #     CONTROL CARDS
              1      DATA TYPE(LOGR) REPORT(YES)
              2      CONTINUE
              3      DEFINE LOGSTREAM NAME(BAD.LOG.STREAM) LOWOFFLOAD(20)
              4              DASDONLY(NO) LS_SIZE(20) STG_DUPLEX(NO)
              5              STRUCTNAME(LISTXX)
              6      DEFINE LOGSTREAM NAME(WILL.BE.CREATED) STG_SIZE(100)
              7              LOWOFFLOAD(20) DASDONLY(YES) HIGHOFFLOAD(90)
              8              STRUCTNAME(LIST01)
              9              MAXBUFSIZE(32768)

ADMINISTRATIVE DATA UTILITY:  MESSAGES                     DATA TYPE = LOGR
         IXG005I LOGR POLICY PROCESSING LINE# 2
         IXG004I LOGR POLICY PROCESSING ENDED WITHOUT ERROR
         IXG005I LOGR POLICY PROCESSING LINE# 3
         IXG018E STRUCTURE LISTXX DOES NOT EXIST
         IXG447I LOGR POLICY PROCESSING FOUND AN ERROR BUT CONTINUES
         RETCODE=00000008 RSNCODE=00000827
         IXG003I LOGR POLICY PROCESSING ENCOUNTERED AN UNEXPECTED ERROR.
         DIAGNOSIS INFORMATION: 00000008 0000F801 05030004 050B000B
         IXG005I LOGR POLICY PROCESSING LINE# 6
         IXG004I LOGR POLICY PROCESSING ENDED WITHOUT ERROR
         IXG446I LOGR POLICY PROCESSING FOUND ERRORS BUT CONTINUED.
         FIRST ERROR FOUND LINE# 3 RETCODE=00000008 RSNCODE=00000827
         TOTAL NUMBER ERRORS FOUND: 1
```

Log Stream Latch Identity (V1R11)

- Logger asked GRS for an ability to associate latches to their resource back in the late 90s when logger was experiencing many latch contention issues
- GRS eventually got around to it, and logger supported it
- Knowing which resources involved in latch contention would be useful to:
  - See what evidence to look for that might have led to the contention
  - Determining what action to take for example:
    - Which jobs to kill
    - Which log streams could be force disconnected
  - Figuring out what the tradeoffs of various actions would be

Log Stream Latch Identity (V1R11) Example

```
SY1 d grs,an,latch,depend,detail
SY1 ISG374I 14.23.53 GRS ANALYSIS
DEPENDENCY ANALYSIS: ENTIRE SYSTEM
----- LONG WAITER #1
         JOBNAME: IXGLOGR (ASID=002B, TCB=005DDE88)
         REQUEST: EXCLUSIVE
LT:7F42907800000001
WAITING 00:01:33 FOR RESOURCE (CREATOR ASID=002B)
SYS.IXGLOGER_LCBIT___CTA:00000003_SLSA:0001
LST:7F42BD00000000BD
2:SOME.LOG.STREAM
         JOBNAME: WRITE3 (ASID=0026, TCB=005D3A08)
         REQUEST: SHARED
LT:7F42901000000002
ANALYSIS ENDED: THIS UNIT OF WORK IS NOT WAITING
```

- In this example system logger (jobname IXGLOGR) is waiting to get the latch exclusive but another job (WRITE3) holds the latch shared

Log Stream Latch Identity (V1R11) Example

- The log stream name can be used to get more information from logger:
-
```
SY1  d logger,c,lsn=SOME.LOG.STREAM,detail
SY1  IXG601I   15.46.06  LOGGER DISPLAY 912
CONNECTION INFORMATION BY LOGSTREAM FOR SYSTEM SY1
LOGSTREAM                  STRUCTURE        #CONN  STATUS
---------                  ---------        ------ ------
SOME.LOG.STREAM            LIST06           000002 IN USE
  DUPLEXING: STRUCTURE
  GROUP: PRODUCTION
  JOBNAME: WRITE3    ASID: 0026
    R/W CONN: 000000 / 000001
    RES MGR./CONNECTED: *NONE*   / NO
    IMPORT CONNECT: NO
  JOBNAME: NJCONC    ASID: 001B
    R/W CONN: 000001 / 000000
    RES MGR./CONNECTED: *NONE*   / NO
    IMPORT CONNECT: NO

NUMBER OF LOGSTREAMS:  000001
```
- A detailed logger display of the log stream connections shows that there are two connectors on this log stream

IXGCNF Parmlib, offload monitoring interval updates (V1R13)

- Logger has monitoring messages for offload data set allocations and recalls and provides default intervals for when they come out, but no tailoring to them
  - `IXG310I SYSTEM LOGGER CURRENT OFFLOAD IS NOT PROGRESSING FOR LOGSTREAM logstream STRUCTURE: strname request DSN=dsnhlq.dsnlsn.dsnllq`
  - `IXG311I SYSTEM LOGGER CURRENT OFFLOAD HAS NOT PROGRESSED DURING THE PAST seconds SECONDS  FOR LOGSTREAM logstream STRUCTURE: strname request DSN=dsnhlq.dsnlsn.dsnllq`
  - `IXG312E OFFLOAD DELAYED FOR logstream, REPLY "MONITOR", "IGNORE", "FAIL","AUTOFAIL", OR "EXIT".`
- For many installations the default values are either
  - Too late – action should have been taken earlier to prevent outage
  - Too early – messages are over-alarming

IXGCNF Parmlib, offload monitoring interval updates (V1R13)

- The IXGCNFxx parmlib member was developed to configure these values
- Syntax:
  - MONITOR OFFLOAD
    - WARNALLOC(xxx)
    - WARNRECALL(xxx)
    - ACTIONALLOC(xxx)
    - ACTIONRECALL(xxx)
  - CTRACE(CTnLOGxx)
- IPL time, SET IXGCNF, SETLOGR operation
- D LOGGER,IXGCNF command

IXGCNF Parmlib, offload monitoring interval updates (V1R13) Example

- Assume the following SYS1.PARMLIB members are defined
  - IXGCNFNN
    - CTRACE(CTILOG01)
  - IXGCNFN1
    - MONITOR OFFLOAD
    - WARNALLOC(5)
    - ACTIONALLOC(15)
  - IXGCNFN2
    - MONITOR OFFLOAD
    - ACTIONRECALL(26)
- IEASYSxx member used during IPL contained IXGCNF=NN
- SET IXGCNF=(n1,n2) was then issued
- SETLOGR MONITOR,OFFLOAD,ACTIONALLOC(20) was then issued

IXGCNF Parmlib, offload monitoring interval updates (V1R13) Example

```
DISPLAY LOGGER,IXGCNF
SY1  IXG607I   18.00.56  LOGGER DISPLAY 395
LOGGER PARAMETER OPTIONS
KEYWORD              SOURCE   VALUE
------------------ -------- ----------------
CTRACE               IPL (NN) CTILOG01
MONITOR OFFLOAD
 WARNALLOC           SET (N1) 00005
 ACTIONALLOC         SETLOGR  00020
 WARNRECALL          DEFAULT  00060
 ACTIONRECALL        SET (N2) 00026
```

Logger Health Checks (OA15593, OA22255)

- Clients wanted to be more aware of resource full conditions without having to read SMF type 88 records
  - Staging data set full
  - Structure element full
  - Structure entry near Full
- Logger health checks provide a listing of these conditions
- OA22255 added PARM('TIME(mm/dd/yyyy hh:mm:ss:)') / PARM('ALL') to tailor output to recent occurrences only

Logger Health Checks (OA15593, OA22255)

```
CHECK(IBMIXGLOGR,IXGLOGR_STAGINGDSFULL)
START TIME: 10/10/2007 13:59:57.386351
CHECK DATE: 20060615  CHECK SEVERITY: LOW
CHECK PARM: TIME(10/10/2007 17:42:34)

* Low Severity Exception *


IXGH008E One of more log streams encountered a staging data set
full condition since 10/10/2007 17:42:34 (GMT).
...
                                                    Time of Last
Log Stream                  Structure        Count Condition (GMT)
TESTLOG1.HCHECK1.S7         STRUCT7          47    10/10/2007 17:42:34
TESTLOG1.HCHECK1.D1         *DASDONLY*       16    10/10/2007 17:44:00
TESTLOG1.HCHECK1.D2         *DASDONLY*       29    10/10/2007 17:45:06
TESTLOG1.HCHECK1.D4         *DASDONLY*       89    10/10/2007 17:46:12


END TIME: 10/10/2007 13:59:57.388904  STATUS: EXCEPTION-LOW
```

Recover From Failures

Get me out of here!

Recover From Failures

- Logger Procs
  - ixgoflds, ixgconls, ixgdells, ixgdelab
- Force Log Stream Disconnect/Delete (V1R7)
- Rename Log Stream (V1R8)

Logger Procs

- Initiate operations against log streams
  - S IXGOFLDS – Start an offload
  - S IXGCONLS
    - Connect to a log stream
    - Issue wtor
    - Disconnect on wtor reply
  - S IXGDELLS – Delete log stream
  - S IXGDELAB – Delete all log stream blocks
- Originally shipped into V1R5, IXGCONLS in OA28150
- Procs in SYS1.SAMPLIB, need to be copied to PROCLIB
- Look for messages IXG273, IXG274, IXG227

Force Log Stream Disconnect/Delete (V1R7)

- Clients wanted an ability to terminate activity on a particular log stream without affecting the whole system or sysplex
- Previous options were as drastic as restarting logger to requiring a sysplex IPL

- SETLOGR FORCE,DISCONNECT
  - Request Logger to force disconnect log streams
  - Log streams hung in ‘Disconnect Pending’ states can be disconnected
- SETLOGR FORCE,DELETE
  - Request Logger to force delete log streams
  - Log streams with failed persistent connections can be deleted

Force Log Stream Disconnect/Delete (V1R7) Example

- Force disconnect on a log stream in disconnect pending state

- D LOGGER,C,LSN=NICKJ.TEST.LOGSTRM1,DETAIL Shows the log stream in Disconnect pending state:

```
IXG601I   12.39.43  LOGGER DISPLAY 769
CONNECTION INFORMATION BY LOGSTREAM FOR SYSTEM SY1
LOGSTREAM                  STRUCTURE        #CONN  STATUS
---------                  ---------        ------ ------
NICKJ.TEST.LOGSTRM1        LIST01           000000 DISCONNECT
  PENDING
  DUPLEXING: STAGING DATA SET
    STGDSN: IXGLOGR.NICKJ.TEST.LOGSTRM1.SY1
            VOLUME=ALL001  SIZE=000300 (IN 4K)   % IN-USE=002
  DISCONNECT PENDING FOR 0004 MINUTES
NUMBER OF LOGSTREAMS:  000001
```

Force Log Stream Disconnect/Delete (V1R7) Example

- SETLOGR FORCE operation is entered.

```
SETLOGR FORCE,DISC,LSN=NICKJ.TEST.LOGSTRM1
IXG651I SETLOGR FORCE DISCONNECT COMMAND ACCEPTED
FOR LOGSTREAM=NICKJ.TEST.LOGSTRM1
IXG661I SETLOGR FORCE DISCONNECT PROCESSED SUCCESSFULLY
FOR LOGSTREAM=NICKJ.TEST.LOGSTRM1
```

- D LOGGER,C,LSN=NICKJ.TEST.LOGSTRM1,DETAIL shows the log stream is no longer connected.

```
IXG601I   12.39.58  LOGGER DISPLAY 801
CONNECTION INFORMATION BY LOGSTREAM FOR SYSTEM SY1
LOGSTREAM                  STRUCTURE        #CONN  STATUS
---------                  ---------        ------ ------
NO MATCHING INFORMATION FOUND.
```

Rename Log Stream (V1R8)

- Logger problem occurs where local fix is to delete and redefine the log stream
- This implies complete loss of log stream data
- Clients requested an ability to fence the old log stream off and make progress on a new instance

- Logger developed the rename function to allow for this.
- IXGINVNT macro and IXCMIAPU DATA TYPE(LOGR)

- UPDATE LOGSTREAM NAME(xname) ...
  NEWSTREAMNAME(xnewstreamname)

Prevent Failures

Don’t make me go through that again!

Prevent Failures

- Dynamic Duplexing Updates (V1R10)
- *Couple Data Set Mismatch Resolution*
  - *Part 1 (V1R7)*
  - *Part 2 (V1R9)*
- *Async IXGWRITE limits*
  - *Unauthorized (OA14125)*
  - *Authorized (V1R11)*
- *Share Options Enforcement (V1R12)*

Dynamic Duplexing Updates (V1R10)

- In z/OS V1R3 Logger provided support for log stream attribute updates
- When log stream attributes are updated, they are acknowledged and marked in a pending state, but are not acted upon until a convenient time such as:
  - Data set switch
  - CF structure rebuild
  - Last disconnection (or first connection) to the log stream in the sysplex
- Log stream duplexing attributes (STG_DUPLEX, DUPLEXMODE, and LOGGERDUPLEX) were only updated during the last disconnection to the log stream in the sysplex

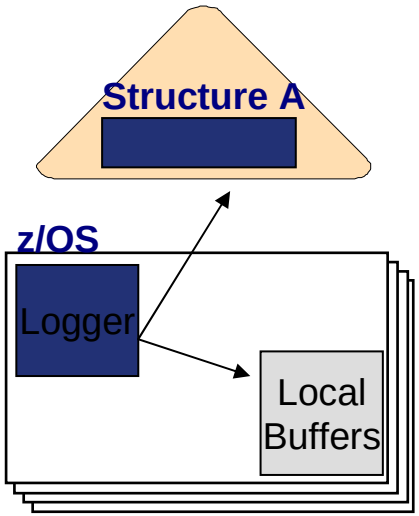
Dynamic Duplexing Updates (V1R10) Duplexing states

Structure Mode / Case #	LOGGER DUPLEX	STG_DUPLEX = NO	STG_DUPLEX = YES	
			DuplexMode = COND	DuplexMode = UNCOND
SIMPLEX				
Case 1	UNCOND/ COND	Local Buffers	Staging DS	Staging DS
Case 2	UNCOND/ COND	Local Buffers	Local Buffers	Staging DS
DUPLEX				
Case 3	UNCOND/ COND	Structure, Local Buffers	Structure, Staging DS	Structure, Staging DS
Case 4	UNCOND/ COND	Structure, Local Buffers	Structure, Local Buffers	Structure, Staging DS
Case 5a	UNCOND	Structure, Local Buffers	Structure, Local Buffers	Structure, Staging DS
Case 5b	COND	Structure	Structure	Structure

Dynamic Duplexing Updates (V1R10)

- Pending updates for log stream duplexing attributes
  - STG_DUPLEX
  - DUPLEXMODE
  - LOGGERDUPLEX
- …will now be effective at the end of a **user managed** structure rebuild

BEFORE
1.a Logger writes to structure and local buffers

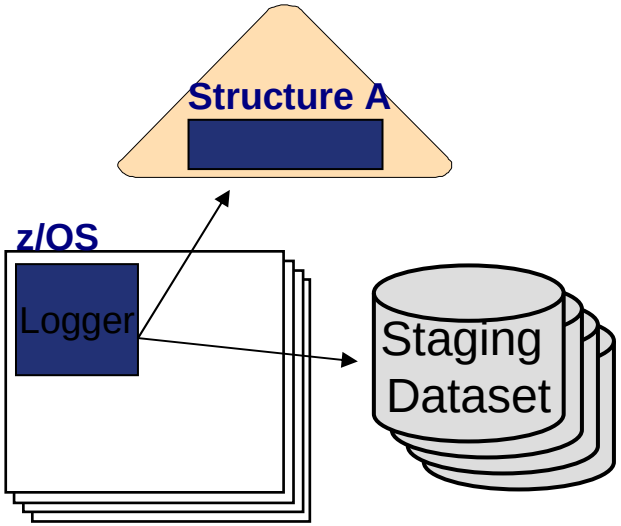


Steps:

a) Update the log stream definition
```
UPDATE LOGSTEAM
STG_DUPLEX(YES)
DUPLEXMODE(UNCOND)
```

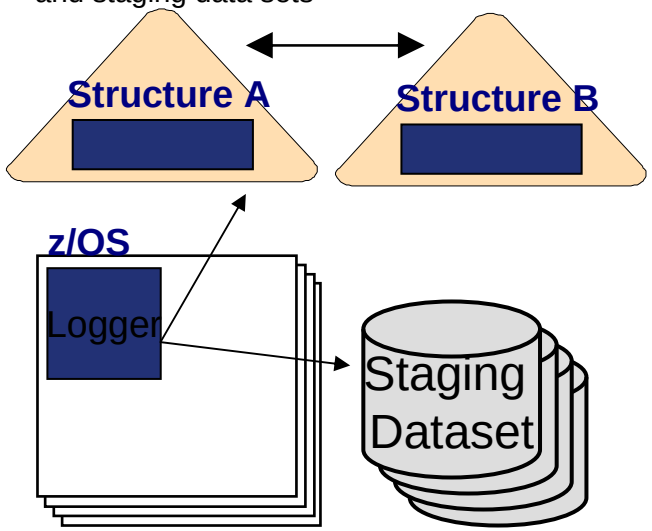
b) Rebuild the structure
```
SETXCF START,REBUILD,
   STRNAME=strname
```

AFTER
1.b Logger writes to structure and staging data sets

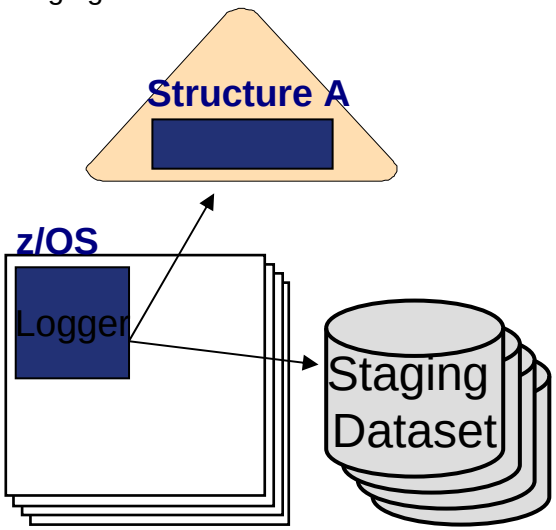


BEFORE

2.a Logger writes to duplex structure and staging data sets



2.b Logger writes to structure and staging data sets



Steps:

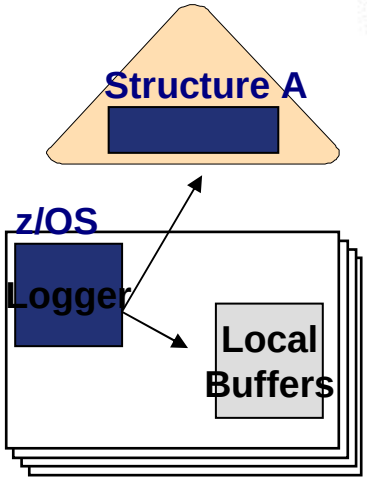
a) Update the log stream definition
`UPDATE LOGSTREAM DUPLEXMODE(COND)`

b) Stop the duplexing rebuild
`SETXCF STOP,REBUILD, DUPLEX,STRNAME=`*`strname`*`, KEEP=OLD`

c) Start a user managed structure rebuild
`SETXCF START,REBUILD, STRNAME=`*`strname`*

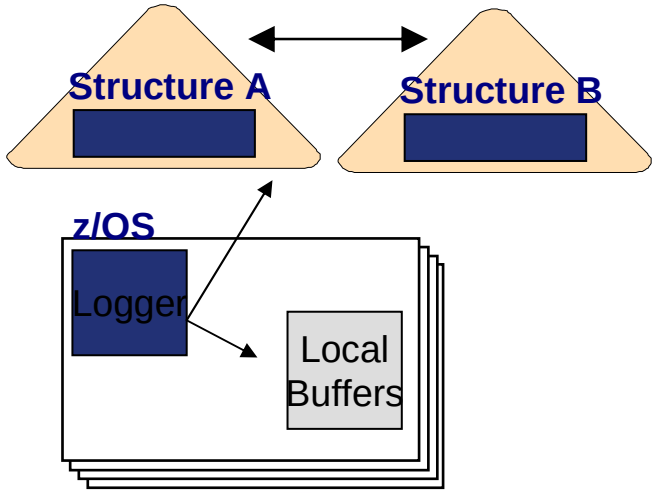
d) Restart the duplexing rebuild if necessary
`SETXCF START,REBUILD, DUPLEX, STRNAME=`*`strname`*

2.c Logger writes to structure and local buffers (A is failure isolated and non volatile)



AFTER

2.d Logger writes to duplex structure and local buffers (A and B are failure isolated and non volatile)



Data set management

Data set management

- Client directed enhancements to data set processing
  - Allow IXGWRITES when structure full (OA36662)
  - Return primary storage consumption info (OA36172)
  - Detect and process deletes during offload thread underway (OA37588)
- Tasking changes help a lot too (see backup)
- *Get off the current offload DS (OA15364)*
- *SVC99 no wait on ENQs for OffloadRecall(NO) (OA20281)*
- *(up to) 4GB log stream data set sizes (OA30548)*

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- A large client running IMS CQS suffered two outages
  - The first was an offload data set allocation hang that prevented offload movement
    - CF storage filled and the log stream was unavailable for an extended period of time
  - The second was a throughput/ bandwith problem where the offload couldn't keep pace caused by an I/O slowdown related to DASD mirroring
    - The offload was progressing but very slowly
    - Structure became full, and logger was unable to accept new IXGWRITES till the offload completed
    - IMS couldn't write or issue a checkpoint (which would write a record and perform deletes that would have freed structure space

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- Part 1: Tuning
- IBM helped client by recommending a supersized log stream approach.
  - Turned the log stream from a funnel type to an active type
  - Made the data more likely to reside in the CF
  - Allowed the client to avoid most dasd movement; all active data in the structure
  - More reaction time in event of dasd movement and problems
- Requires APAR OA36261- Log streams in CF structures larger than 4GB were offloading sooner than expected

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- Part 2: service corrections
  - OA36175 - On CF structure full: offload lesser amount vs. all the way to LOWOFFLOAD value
  - OA36662 - IxgWrite allowed when structure full for authorized callers
  - OA36172 - IxgWrite returns primary storage consumption info (trigger points)
  - OA37588 - Recognize IXGDELET low valid point change during offload

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- OA36175
  - When the log stream structure becomes full, instead of offloading down to LOWOFFLOAD, offload to 90%
  - Client had a low offload of 0% during problem
  - Significant less log data needs to be moved to dasd before applications can resume writing

100%	
90%	
High offload	
Low offload	
0%	

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- OA36662
  - IXGWRITE operations allowed after structure full encountered for authorized logger exploiters
  - Logger no longer halts on the front end of the IXGWRITE operation when the structure was full for authorized callers
    - Log stream writer can try and writes will succeed as soon as any space in the structure is freed
  - IMS APAR supported this and allows them to write checkpoint data sooner and issue delete requests
  - See ICN 1197 (see appendix A)

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- OA36172
  - IXGWRITE returns primary storage consumption usage (trigger points) in the IXGANSAA area
  - Trigger points reveal how much of the primary storage is currently in use
  - IMS apar uses this information to issue extra checkpoints before the structure fills
  - See ICN 1197 (Appendix B for full details)

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- IXGANSAA - ANSAA_WRITETRIGGERSRETURNED (bit) when on
  - On IXGCONN – means that write trigger data will be returned on successful IXGWRITE requests
  - On IXGWRITE – means that write trigger data has been returned
- IXGANSAA - ANSAA_SERVICESPECIFIC
  - ANSAA_GAPS_NEXT_BLKID encapsulated into ANSAA_IXGDELET section of union
  - ANSAA_WRITETRIGGERS added with ANSAA_IXGWRITE section of union
- ANSAA_STRUCTUSEPERCENT – percent of structure in use
- ANSAA_STAGINGUSEPERCENT – percent of staging ds usr

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- ANSAA_WRITEFLAGS – how much primary storage is in use
  - ANSAA_WRITEABOVEHIGHOFFLOAD – this IXGWRITE was over HIGHOFFLOAD
  - ANSAA_WRITEELEVATEDCAPACITY – this IXGWRITE over 1/3 between high and full
  - ANSAA_WRITEIMMINENTCAPACITY – this IXGWRITE over 2/3 between high and full

- IXGQBUF - QBUF_LS_OFFLOAD_RETURNED
  - On when HIGHOFFLOAD and LOWOFFLOAD returned
- QBUF_LS_HIGHOFFLOAD
  - Logstream HighOffload percentage
- QBUF_LS_LOWOFFLOAD
  - Logstream LowOffload percentage

Recent Development APARs (OA36175, OA36662, OA36172, OA37588)

- OA37588
  - Logger Offload processing will be enhanced to recognize IXGDELETs that occur during offload processing at key points
    - After the initial ‘Fast Delete’ Phase
    - After a data set switch
  - Logger will then
    - Check to see if data can be deleted from primary storage and hasn’t been moved to DASD yet
    - Process the ‘fast delete’
    - Continue offloading to DASD if necessary

Backup Charts

- Tasking improvements
- Prevent Failures
- Data set management
- DASDONLY log stream offloads
- CF log stream offload process
- Duplexing Support and List Structure Rebuild
- Some system programmer hints/tips
  - Implementation Activities
    - Read and Plan
    - Setup Authorization
    - Format CDS
    - Update Parmlib members
    - Operations / Commands
    - Common Setup Issues
  - Appendices / additional references

Backup Charts

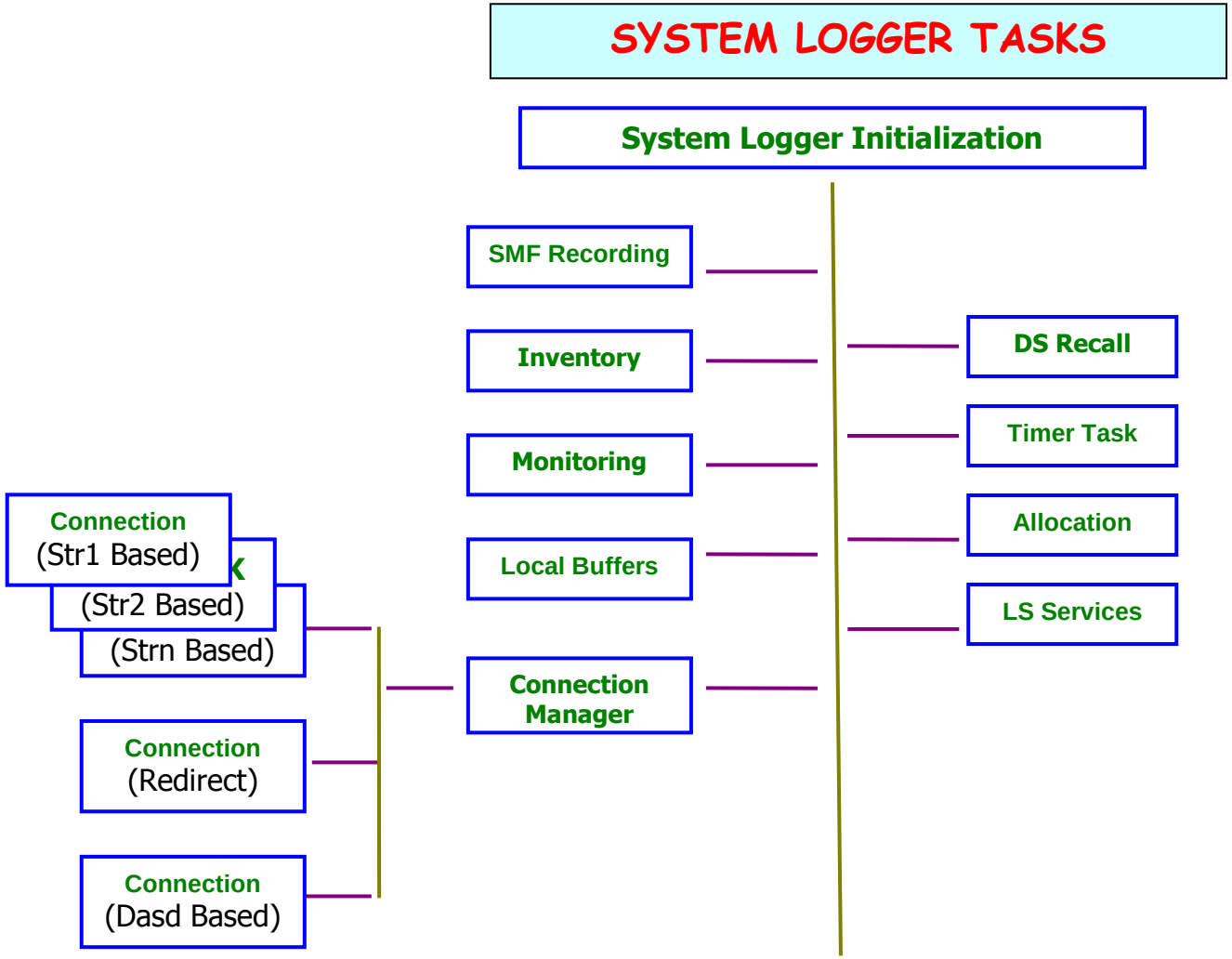
Tasking improvements

Get out of your own way!

Tasking Improvements

- >1024 Dasdonly Log Streams (V1R7)
- Group PROD and TEST log streams (V1R8)
- Asynchronous Data Set Recall (V1R9)
- MTTR enhancements (V1R11)

V1R6 Task Structure

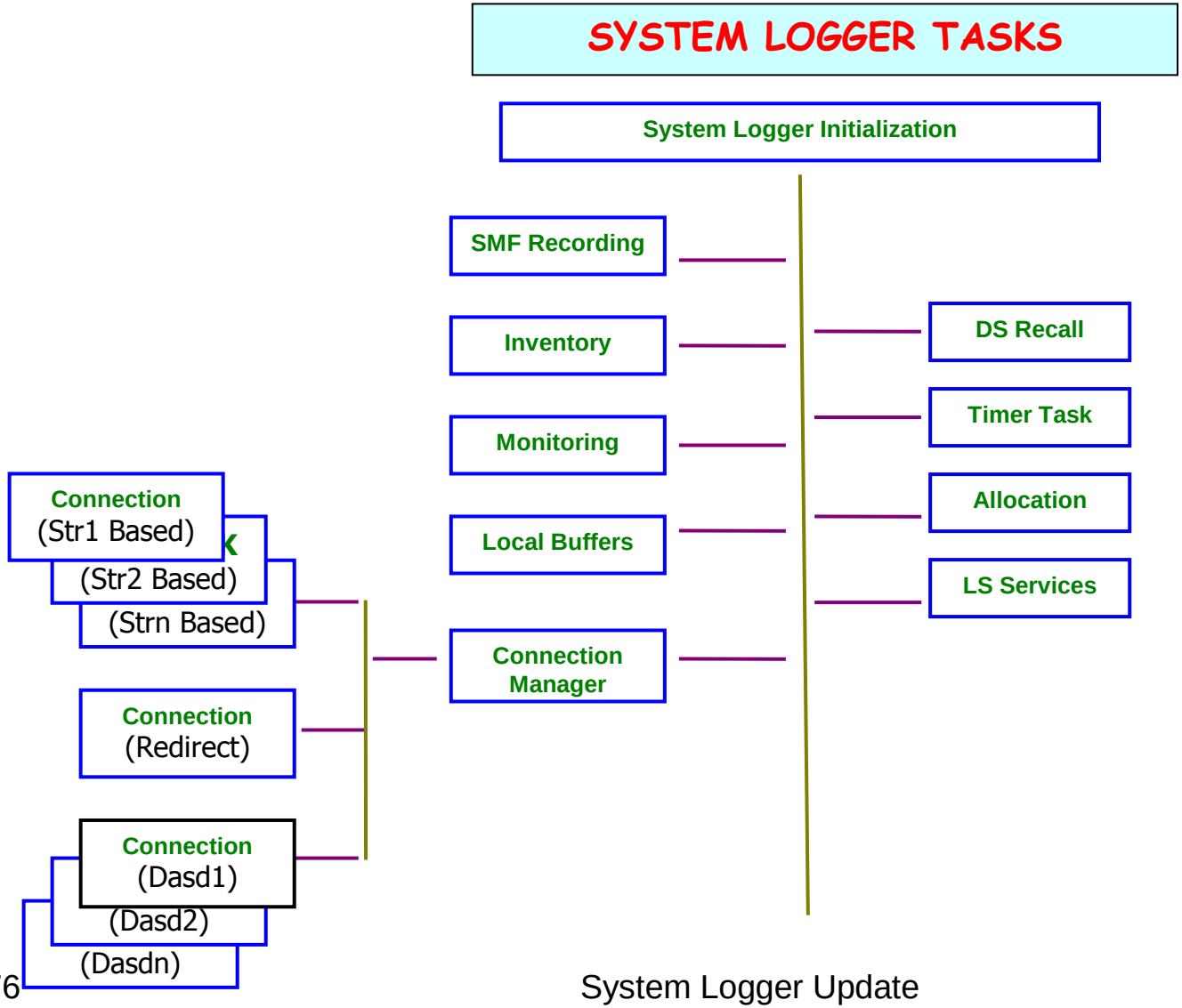


>1024 DASDONLY Logstreams (V1R7)

- Only 1024 DASDONLY log streams at a time were supported
- Logger only used one task to manage these, a failure in that task signaled failure of all DASDONLY log streams

- Task structure was redesigned to allow up to 256 tasks handle DASDONLY log stream activity
- Logger now supports up to 16,384 dasdonly logstream connections simultaneously (up to 64 per task)

V1R7 Task Structure



Group PROD and TEST log streams (v1R8)

- System Logger installations combine both test and production work in the same sysplex.
  - Possible for work on test log streams to have an adverse effect on production log streams.
  - Problems encountered on a test log stream can lead to outages effecting production log streams

- Some Logger tasks, such as data set recall, and data set allocation are single threaded
  - Data set recall for the test log stream can hold off the data set recall for production log stream

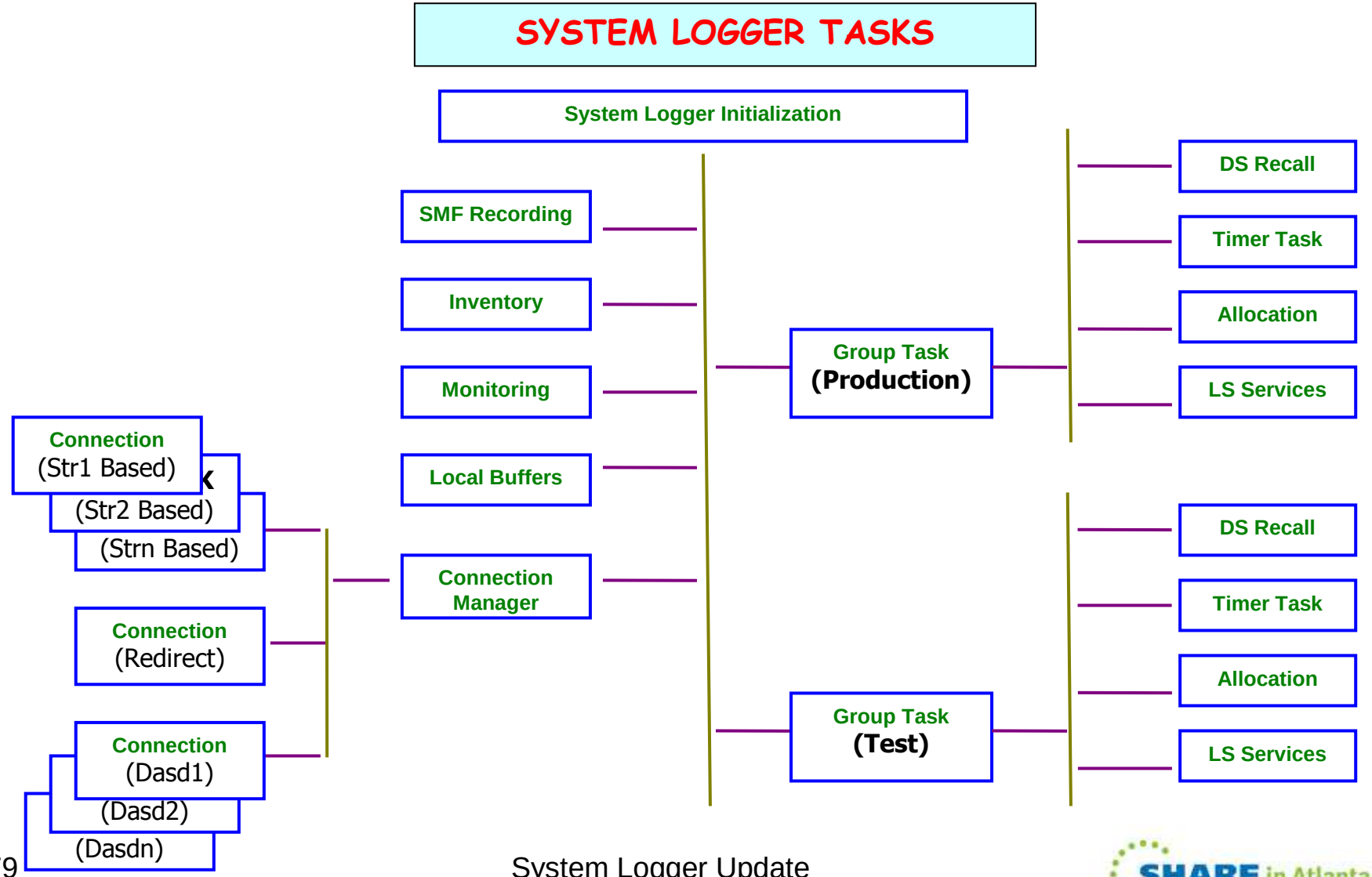
- Sharing of resources
  - LOGR couple data set extents

Group PROD and TEST log streams (v1R8)

- Allow installation to define test and production log streams so that there is separation of work between the two sets
  - Focus of logger tasks
    - Separate Allocation Task
    - Separate Data Set management Task
  - Limitation of resources used by test log streams
    - 25% resource allocation for TEST log streams
    - Ability for production log streams to use more than 75% of resources
- *DEFINE LOGSTREAM ...*

```
STG_DUPLEX(NO),
GROUP(TEST)     ! Or PRODUCTION
```

V1R8 Task Structure



Asynchronous Data Set Recalls (V1R9)

- Logger had single threaded, synchronous handling of recall requests for migrated log stream data sets.
- Means each data set recall must be satisfied (successfully or otherwise) before the next migrated data set recall is requested.
- Occurs in 2 tasks; one for PRODUCTION group, and one for TEST group.
- Can cause limited or slower access to the log stream resource.
  - e.g. A recall request could be necessary during a log stream's offload activity, or when an application is browsing log data.
- Can result in interference between different log stream activities.
  - e.g. A recall request for one log stream data set could hold up a recall request for another log stream data set.

Asynchronous Data Set Recalls (V1R9)

With V1R9 logger provided:

- log stream data set asynchronous recalls:
  - Allows for multiple concurrent, migrated data set recalls
  - DFSMShsm or equivalent function
- Ability to display data sets being recalled by Logger
- Ability to have Logger stop waiting on a data set recall

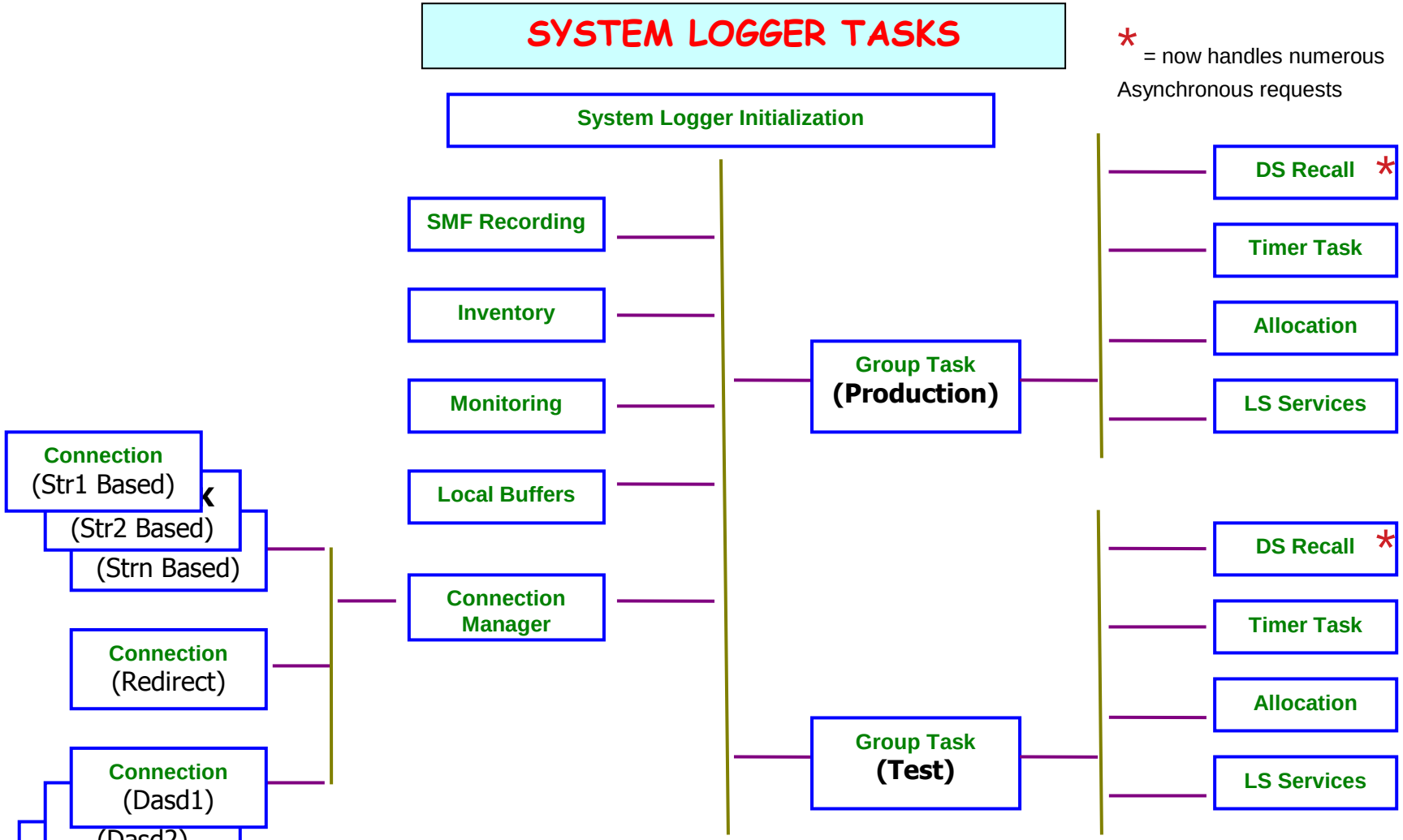
Asynchronous Data Set Recalls (V1R9)

- See messages IXG278, IXG279, IXG280, and IXG281
- `DISPLAY LOGGER,STATUS,RECALLS`

```
IXG601I hh.mm.ss LOGGER DISPLAY [id]
SYSTEM LOGGER STATUS
SYSTEM    SYSTEM LOGGER STATUS
SY1       ACTIVE
          LOGGER DATA SET RECALLS
           GROUP:  PRODUCTION
            SECONDS   DATA SET NAME
            00000038  IXGLOGR.PROD.STREAM01.A0000000
            00000173  IXGLOGR.PROD.STREAM02.A0000003
           GROUP:  TEST
            SECONDS   DATA SET NAME
            00000005  IXGLOGR.TEST.STREAM08.A0000722
```

- `SETLOGR FORCE,NORECALL,DSN=hlq.log.stream.suffix`
  - Allows operator to stop recalling a problem data set

V1R9 Task Structure



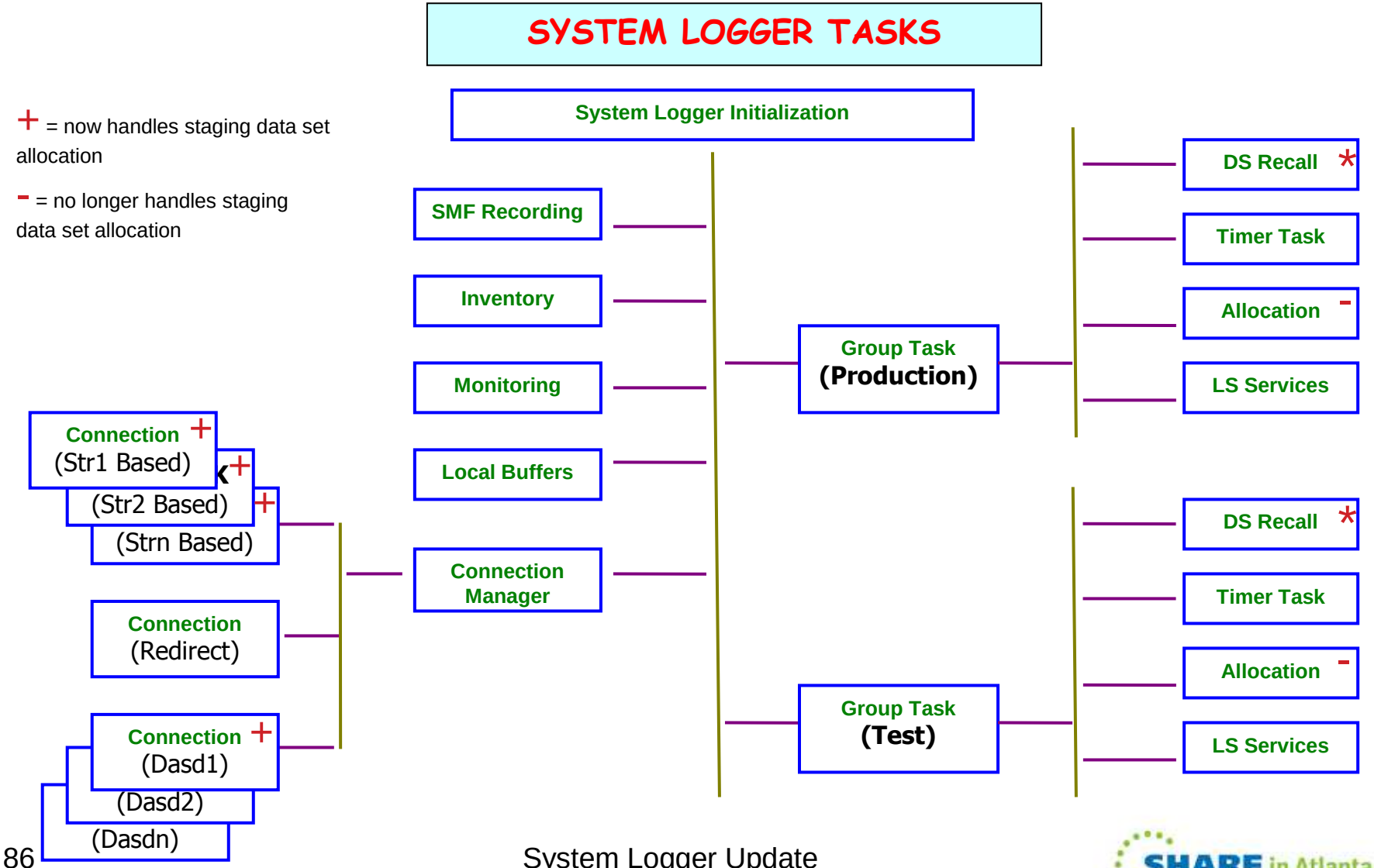
MTTR enhancement (V1R11)

- Logger staging data sets provide a recoverable copy of log data when it is initially written into a log stream.
- A staging data set is allocated on the first connection to a log stream from that particular system in the sysplex.
- Prior to z/OS v1r11:
  - all staging data sets on one system were allocated under one logger subtask
  - very competitive as work started & exploiters connected to their log streams
  - resulted in sequential handling of allocation and data set prep

MTTR enhancement (V1R11)

- Solution:
  - Allocate each staging data set under its respective log stream connection task
  - Reduce general path length for any log stream connection
- Benefit:
  - Removing single task bottleneck allows for more concurrent staging data set allocations
  - Results in much faster turn around for large amounts of simultaneous log stream connections, especially when staging data sets are used

V1R11 Task Structure



Prevent Failures

Don’t make me go through that again!

Couple Data Set Mismatch Resolution parts 1 (V1R7) and 2 (V1R9)

- Problems exist when clients brings in a LOGR couple data set but there is mismatched information such as unexpected used entries in in the CF List Structures
- These lead to numerous undetermined errors, that may appear long after an IPL
- We have several possibilities here:

		CDS Log Stream Connection State		
		Never Connected	No Current Connections	Current Connections
CF State	Empty	ok	ok	missing data
	Good Data	n/a	n/a	ok
	Bad Data	Part 1	Part 2	Problem 3

Couple Data Set Mismatch Resolution parts 1 (V1R7) and 2 (V1R9)

- Problem 1: A never connected log stream finds data in the CF
  - As part of force log stream support, there was a possibility of leaving data in the structure lists if a disconnect did not succeed and was forced.
  - A new – never connected log stream, cleans the list entries before use to avoid problems
- Problem 2: A log stream with no current connections finds data in the CF.
  - In V1R9 the state view was corrected to treat this scenario as the never connected case.
  - Logger will clean out the structure list entries if a log stream connection is requested when there are currently no log stream connections

Couple Data Set Mismatch Resolution parts 1 (V1R7) and 2 (V1R9)

- Problem 3: A log stream with current connections sees data in the CF.
  - With this scenario, the default action is for logger to use the data in the CF.
  - This is ok if the CF data matches the CDS defined log stream
  - This could lead to drastic failures if the CDS and CF mismatch
    - Data in the CF could be newer than data in the CDS
    - Data in the CF could be for another log stream or entirely different CF exploiter
  - Logger may try to offload the data into the defined log stream data sets, and this could fill the log stream with bad data, or more likely, the data doesn't conform to the log stream's expectations. Logger then could abend at the next log stream offload, which could be much later than the IPL.
  - No current solution, other than operator vigilance

Async IXGWRITE limits (V1R11, OA14125)

- Logger exploiters issuing IXGWRITE requests with keywords
  - MODE=SYNCECB
  - MODE=SYNCEXIT
  - MODE=ASYNCNORESPONSE
- Often don't have control over the frequency of their requests and may write **a lot** more during workload spikes
- IXGWRITE requests that drive asynchronous logger processing, cause logger to copy and maintain many buffers for log data processing.
- At some point the dam breaks, logger has overused the systems storage pools, and the system likely wait states
- Logger needed to prevent log stream writers from killing the system

Async IXGWRITE limits (V1R11, OA14125)

- Logger now limits concurrent asynchronous IXGWRITES per log stream connection to
  - 2000 writes for unauthorized writers
  - 10000 writes for authorized writers
- Logger will prevent IXGWRITES with return code 8 reason code 867
  - ANSAA_DIAG1 has a 1 for unauthorized callers
  - ANSAA_DIAG1 has a 2 for authorized callers
  - ANSAA_DIAG2 has the total number of outstanding IXGWRITE requests for this connection

Share Options Enforcement (V1R12)

- A common user error was the installation not setting share options for system logger data sets
  - Logger requires sharing of log stream data sets for the use of log streams across multiple data sets
  - To facilitate this SHAREOPTIONS(3,3) is required for log streams that have a potential multi system access
  - Not setting share options for system logger data sets when required, can lead to the following problems:
    - Data set access errors browsing log data
    - Data set access errors offloading log data
    - Errors using staging data sets to recover from a failed connection
    - Leads to application outages; extremely problematic for exploiters who can not tolerate loss of data

Share Options Enforcement (V1R12)

- System Logger now corrects the VSAM SHAREOPTIONS for NEW log stream data sets with share options less than (3,3) when required
- Message IXG282I indicates that share options have been changed
- LIST LOGSTREAM NAME(*logstreamname)* DETAIL(YES) **LISTCAT** option added to show IDCAMS LISTCAT ALL output for logger offload data sets.

Data set management

Get off the current offload DS (OA15364)

- Logger was unable to delete eligible offload data sets when systems other than the deleting system had an old data set allocated as the current
- This leaves the data set around eating up space in the data set directory
- The data set directory filling and exhaustion of data set extents could lead to an outage for the log stream (or another log stream)
- At the end of the offload logger now
  - Checks on each connected system if it is on the real current data set
  - Deallocates the data set if it is no longer current
  - Before data set deletion phase of offload processing

SVC99 no wait on ENQs for OffloadRecall(NO) (OA20281)

- Logger attempts to allocate the current offload data set on SYSB via SVC99 request
- SYSA is holding the current offload data set and is in the progress of migrating the data set and holds the ENQ
- The log stream definition specifies OffloadRecall(NO) but Logger on SYSB waits a significant time for the migration to complete before eventually obtaining a new offload data set

- Specifying NOWAIT on the SVC99 request will help logger avoid this type of delay

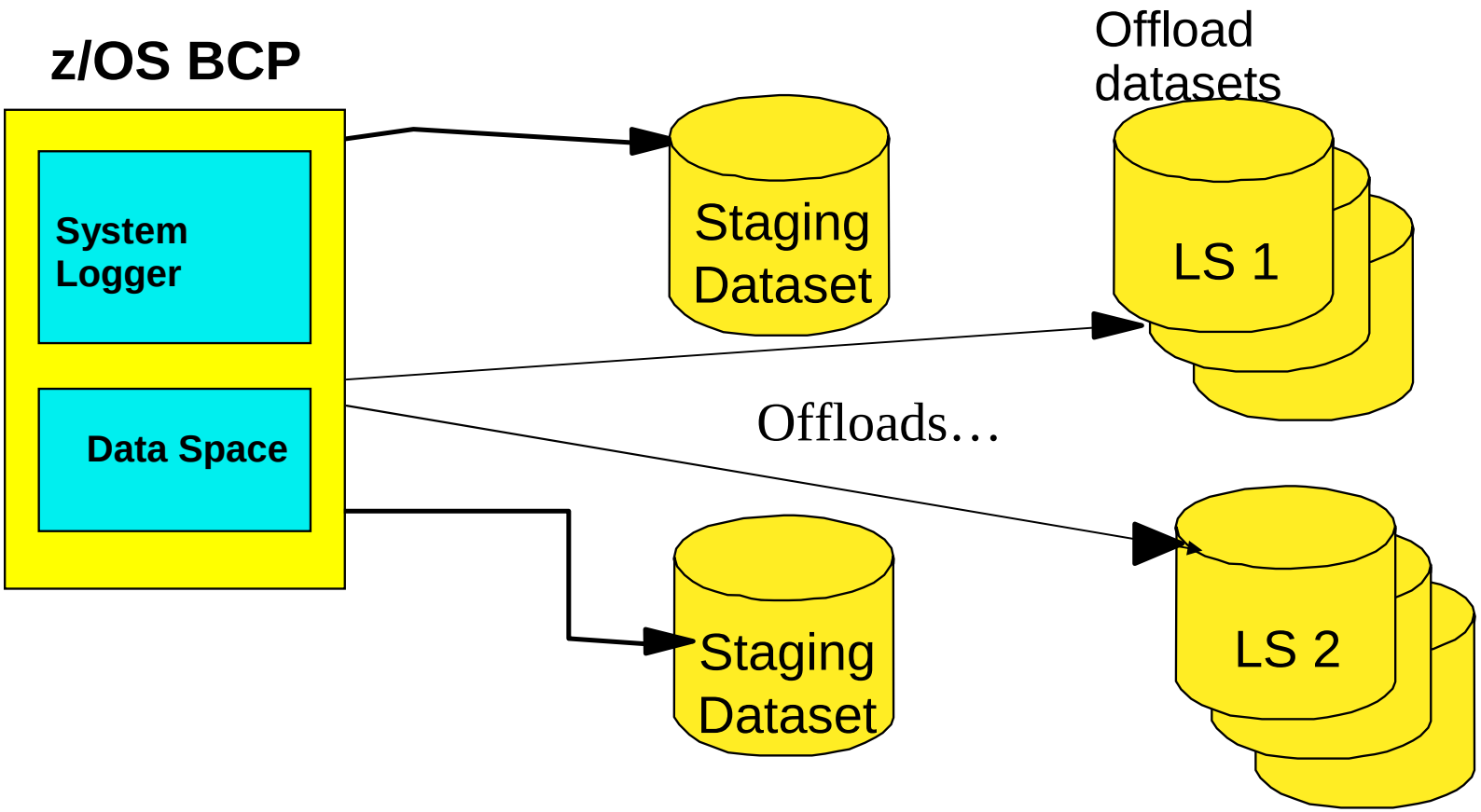
4GB log stream data set sizes (OA30548)

- When logger was introduced there was a 2GB limit on CF size
- This implied a 2GB limit on staging data set sizes, and logger had represented size fields as signed 4 byte integers
- OW44389 (2000) introduced logic to prevent data sets from going over 2GB. If data sets are greater than 2GB the data set will be deleted and allocated at 2GB.
  - Over 2GB meant overflow…
- The 2GB limit has since been abolished and XES allows for structures much larger than 4GB but the 2GB data set size limit remained

4GB log stream data set sizes (OA30548)

- With OA30548, logger introduced support for data set sizes up to 4GB in size
- Requires toleration APAR OA31461
  - To correct the signed to unsigned fields…
- OA37972 recently corrected issues with OA30548
  - Clients with data classes that allow for greater than 4GB data sets were allocating data sets greater than 4GB but logger was only using a fraction
  - Logger will now detect the data set is over 4GB and try to get a data set under (near 3.5 GB)
- Still more work to do… we would like to someday support greater than 4GB data sets

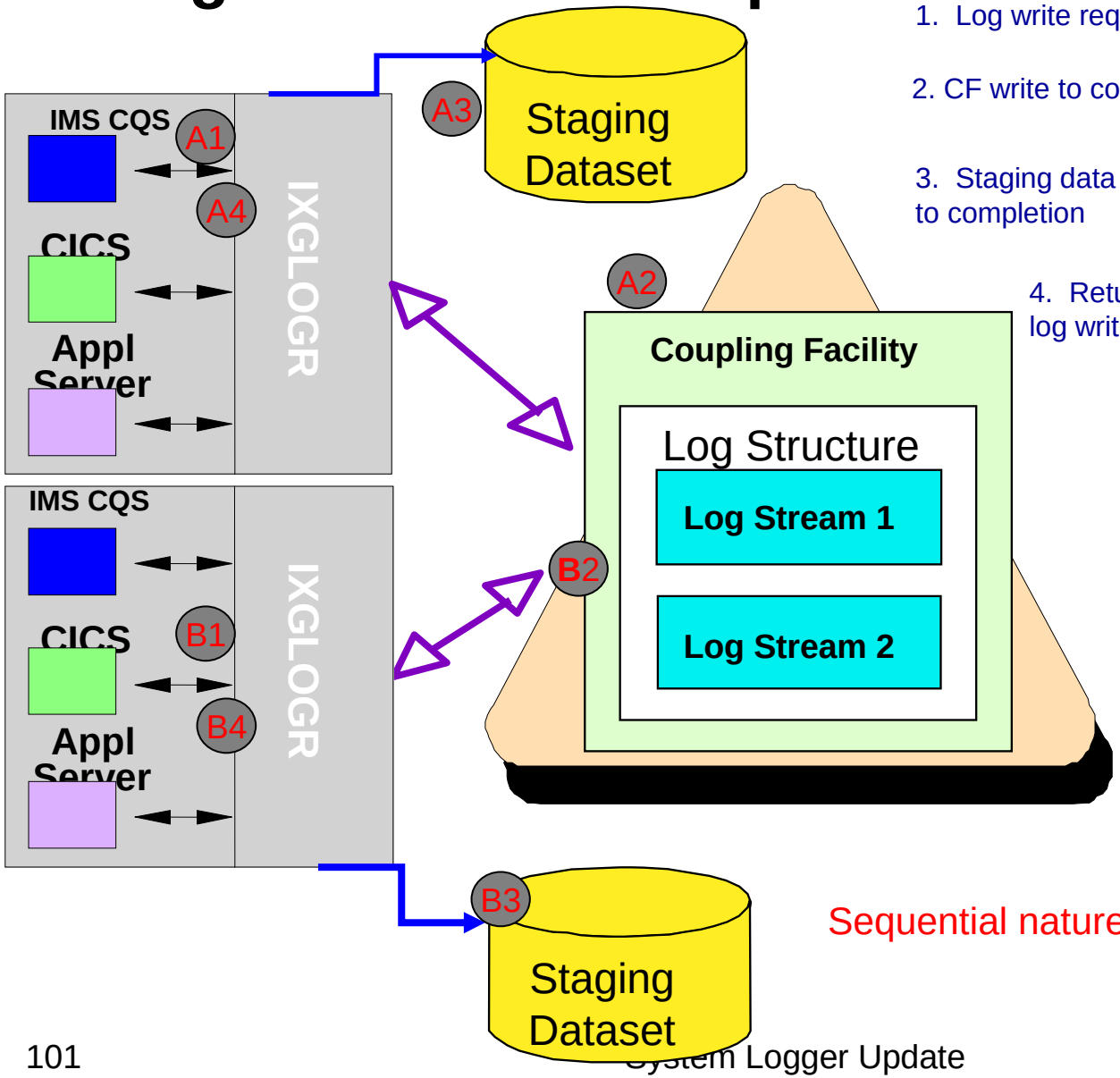
DASDONLY log stream offloads



------- Primary storage log stream data ----

-- Secondary storage log stream data --

CF log stream offload process

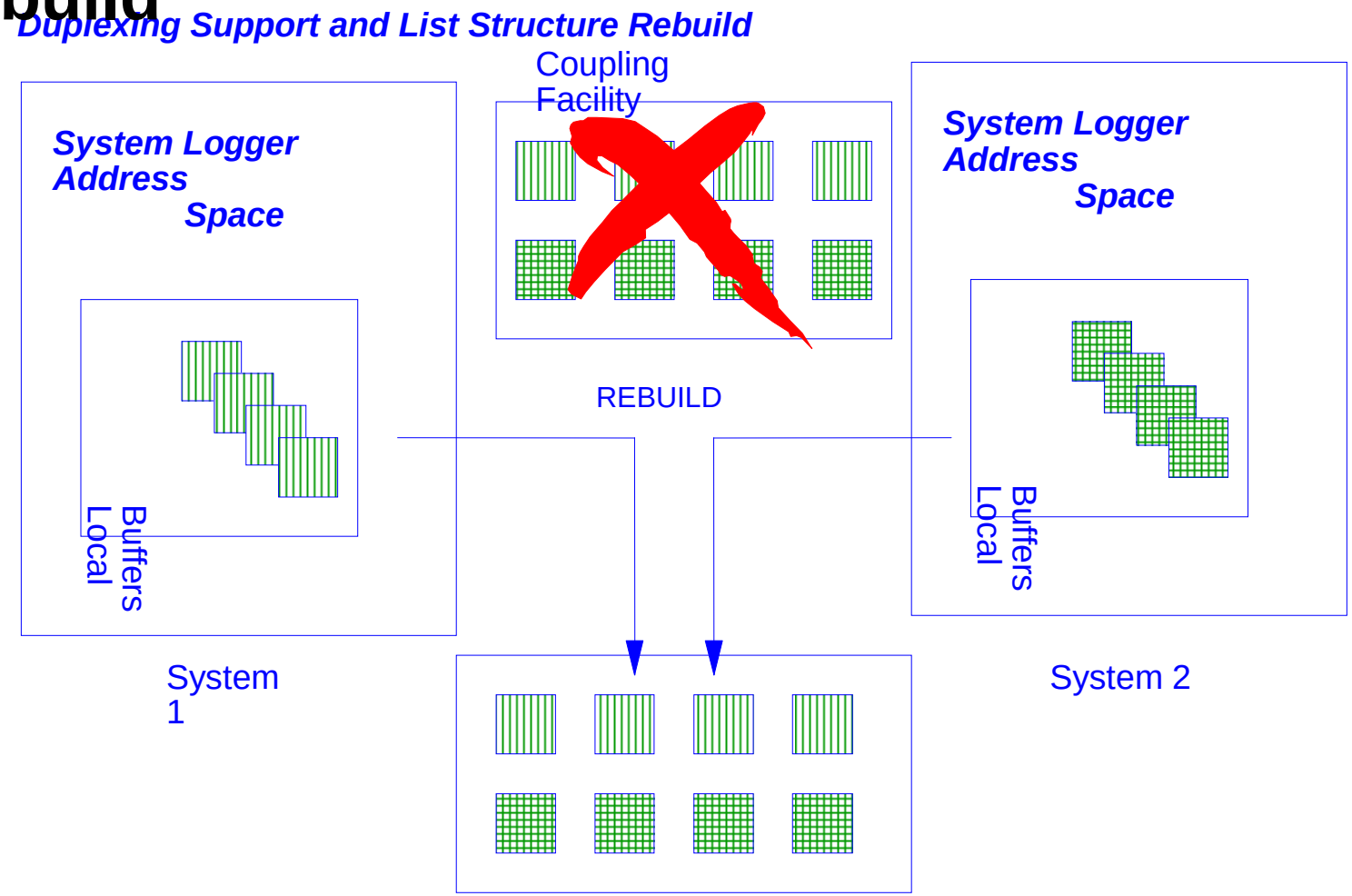


1. Log write requests
2. CF write to completion
3. Staging data set write to completion
4. Return to log writer

A1 B1
A2 B2
A3 B3
A4 B4

Sequential nature of log data writes

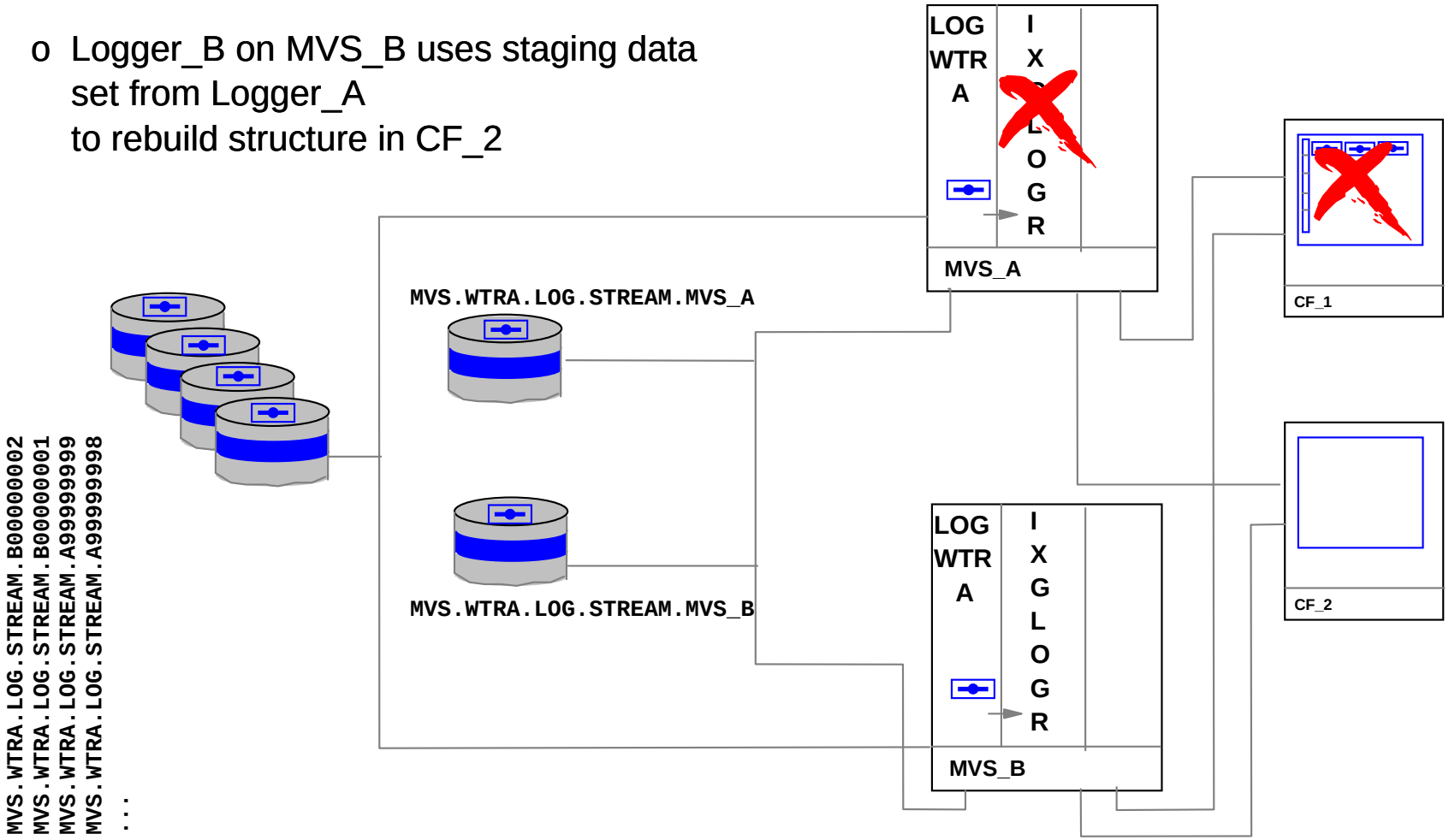
Duplexing Support and List Structure Rebuild



No single point of failure. DASD may also be duplexed.

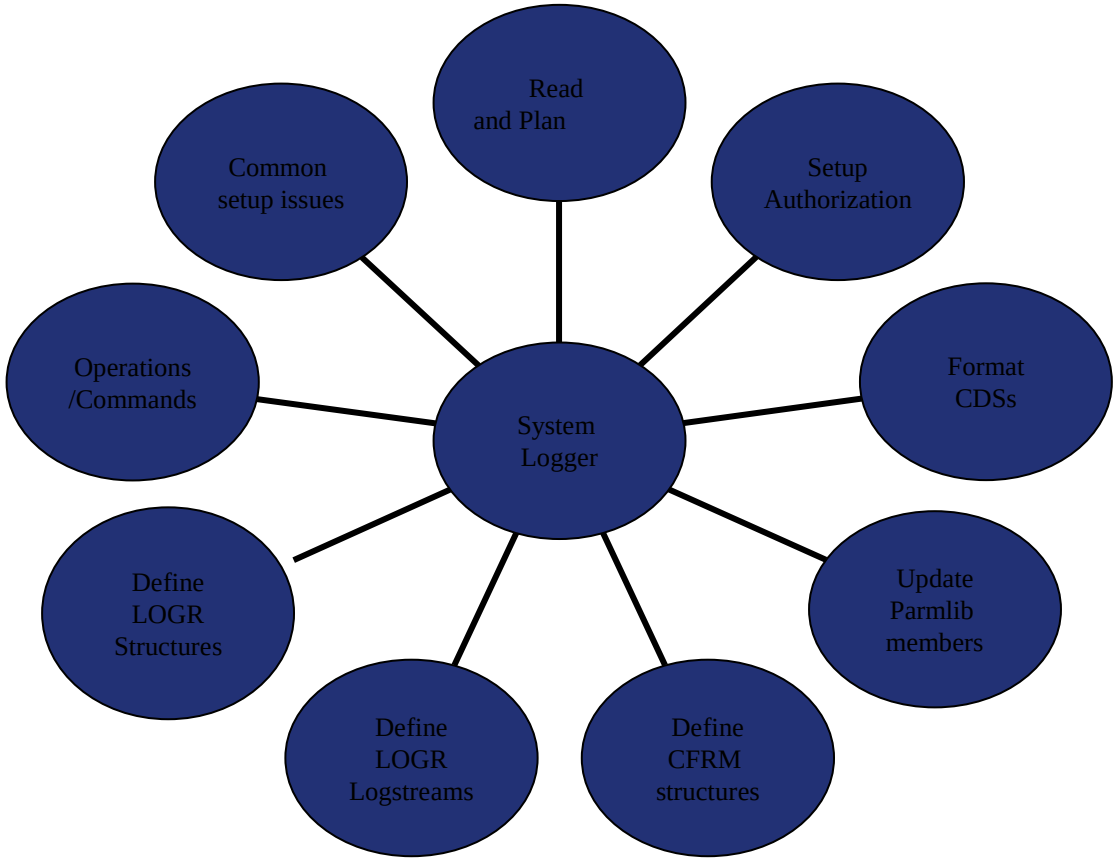
REBUILD processing accomplished through local buffers

- Failure of MVS_A and CF_1
- Logger_B on MVS_B uses staging data set from Logger_A to rebuild structure in CF_2

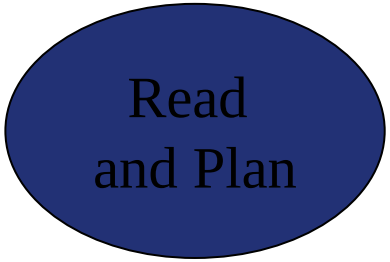


Some system programmer hints/tips

Implementation Activities



Read and Plan



First, see Appendix B for some good reading materials

Then plan Logger configuration, highlights:

- Couple Data Sets (CDSs)
- CF structures (sizes and connectivity)
- DASD space, access, naming conventions, and migration scheme

Setup Authorization

Setup Authorization

Provide access for:

- System Logger address space
- CRFM and LOGR CDSs
- APIs to logstreams and structures

RACF Definitions -

For System Logger Address Space

- IBM recommends that the System Logger address space (IXGLOGR) be assigned privileged and/or trusted RACF status.

- If the System Logger address space for your installation is neither privileged nor trusted, make sure to give IXGLOGR the following SAF authorizations:
  - Define IXGLOGR in RACF started procedures table (SPT) or CLASS(STARTED)
  - For coupling facility logstreams, define alter access to RESOURCE(IXLSTR.structure_name) CLASS(FACILITY) for access to the logstream coupling facility structures
  - Define alter access to RESOURCE(hlq.data_set_name) CLASS (DATASET) for each DASD logstream and stagingdataset
  - Define read access to RESOURCE(sys1.parmlib_data_set_name) CLASS (DATASET) for access to SYS1.PARMLIB

RACF Definitions -

For CFRM, Inventory and API

- Authorize who can use IXCMIAPU
  - Define a resource profile for the resource name MVSADMIN.LOGR to the FACILITY class. Assign ALTER access authority to users who must alter or maintain the policy; assign READ access authority to users who require reports on the policy, but who will not change the policy.
  - If applicable, define a resource profile for the resource name MVSADMIN.XCF.CFRM to the FACILITY class. Assign UPDATE access to users who must define coupling facility structures in the policy.

- Specify ALTER access authority to RESOURCE(log_stream_name) CLASS(LOGSTRM) to DEFINE, UPDATE, or DELETE LOGSTREAM.

RACF Definitions –

For CFRM, Inventory and API cont

- Specify ALTER access authority to RESOURCE(IXLSTR.structurename) CLASS(FACILITY) to DEFINE or DELETE STRUCTURE, .

- Specify UPDATE access authority to RESOURCE(IXLSTR.structurename) CLASS(FACILITY) to allow the specification of a structure name on a LOGSTREAM definition.

For example:

DEFINE LOGSTREAM(log_stream_name)

STRUCTNAME(structname)

Format CDS

Format CDSs

IXCL1DSU format utility

- SYSPLEX CDS
- CFRM CDS
- LOGR CDS
- recommended:
  have alternate formatted/active
  and also a spare one ready too

Format LOGR CDS

- Format the MVS Logger Inventory Couple Data Set with:
  - LSR - Number of logstreams
  - LSTRR - Number of structures
  - DSEXTENT - Number of dataset directory extents

```
//STEP1    EXEC PGM=IXCL1DSU
//SYSPRINT DD *
//SYSIN    DD *
      DEFINEDS SYSPLEX(PLEX1)
           DSN(PRIMARY.LOGR.DATASET) VOLSER(Y43SD5) CATALOG
      DATA TYPE(LOGR)
           ITEM NAME(LSR) NUMBER(15)
           ITEM NAME(LSTRR) NUMBER(15)
           ITEM NAME(DSEXTENT) NUMBER(5)
           ITEM NAME(SMDUPLEX) NUMBER(1)
           . . .
      DEFINEDS SYSPLEX(PLEX1)
           DSN(ALTRNATE.LOGR.DATASET) VOLSER(Y43APC) CATALOG
           . . .
```

Update Parmlib members

Update Parmlib members

SYS1.PARMLIB members:

- IEASYSxx parms
- COUPLExx
- GRSRNLxx
- IEFSSNxx
- others as required, e.g. CONSOLxx

SYS1.PARMLIB - COUPLExx

- Update COUPLExx for Logger Couple Data Sets
  - DATA TYPE(LOGR)
    PCOUPLE(primary-dsname[,primary-volume])
    ACOUPLE(alternate-dsname[,alternate-volume])
  - also include sysplex CDS and CFRM CDS

Activate the parmlib updates:

- IEASYSxx COUPLE=xx
- or use commands:
- SETXCF PCOUPLE,TYPE=LOGR,DSN=(primary.logr.cds,volser)
- SETXCF ACOUPLE,TYPE=LOGR,DSN=(alternate.logr.cds,volser)

SYS1.PARMLIB - GRSRNLxx

- Update GRSRNLxx for Logstream Data Sets:

```
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSDSN) RNAME(IXGLOGR)
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSDSN) RNAME(hlq)
```

- Notes:
  - (1) QNAME is the major name of the resource - SYSDSN
  - (2) RNAME is the minor name of the resource
    - IXGLOGR is the default high level qualifier for logstream datasets

- Activate the parmlib updates:
  - IEASYSxx GRSRNL=(xx,xx,...)
  - SET GRSRNL=(xx,yy,…)

SYS1.PARMLIB - IEFSSNxx

- Update IEFSSNxx to activate the LOGR subsystem for ("browse interface“) access method:
  - SUBSYS SUBNAME(LOGR) INITRTN(IXGSSINT)
- Activate the parmlib updates:
  - IEASYSxx SSN=(xx,yy…)
  - SETSSI ADD,SUBNAME=LOGR,INITRTN=IXGSSINT

(see sys1.samplib IXCCFRMP)

IXCMIAPU

Administrative Policy Utility

Define CFRM structures

CFRM CDS (Policy1)

```
CF   NAME(FACIL01)
        ......
 STRUCTURE  NAME(IXCSTR1)
        …..
CF   NAME(FACIL02)
        ......
 STRUCTURE   NAME(LOGGER_LOGREC)
 STRUCTURE   NAME(LOGGER_OPERLOG)
```

```
//STEP1    EXEC PGM=IXCMIAPU
//STEPLIB  DD   DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
    DATA TYPE(CFRM)
    DSN(SYS1.CFRM.CDS01)  REPORT(YES)
      DEFINE POLICY NAME(POLICY1}
       CF NAME( FACIL02)
          .......
       STRUCTURE NAME(LOGGER_LOGREC)
       STRUCTURE NAME(LOGGER_OPERLOG)
```

(see sys1.samplib IFBLSJCL)

IXCMIAPU

Administrative Policy Utility

Define LOGR logstreams

Define LOGR structures

LOGR CDS (inventory)

logger definitions:
.....
LOGSTREAM
NAME(SYSPLEX.LOGREC.ALLRECS)
.....
STRUCTURE
NAME(LOGGER_LOGREC)

```
//STEP1    EXEC PGM=IXCMIAPU
//STEPLIB  DD   DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
    DATA TYPE(LOGR)
    DSN(SYS1.LOGR.CDS01)  REPORT(YES)
     DEFINE STRUCTURE NAME(LOGGER_LOGREC)
     DEFINE LOGSTREAM
            NAME(SYSPLEX.LOGREC.ALLRECS)
            STRUCTNAME(LOGGER_LOGREC)
```

Log stream and data set names

- Define Logstream an example
  - NAME(DBDC.CICSVR.DFHLGLOG)
  - EHLQ(CICS.CICSTS23)
    - Offload dataset name(s):

      CICS.CICSTS23.DBDC.CICSVR.DFHLGLOG.*Axxxxxxx*
    - Staging dataset name(s):

      CICS.CICSTS23.DBDC.CICSVR.DFHLGLOG.*sysname*
      for CF based logstreams

      CICS.CICSTS23.DBDC.CICSVR.DFHLGLOG.plexname
      for DASDONLY logstream

Operations / Commands

Operations /Commands

Logger address space:

D Logger,STatus [,options] Logger address space status info
D J,IXGLOGR

FORCE IXGLOGR,ARM
S IXGLOGRS

LOGR CDS related commands (XCF):

D XCF,COUPLE,TYPE=LOGR
SETXCF COUPLE,TYPE=LOGR,PCOUPLE=(...),ACOUPLE=(...)
SETXCF COUPLE,TYPE=LOGR,PSWITCH

Operations /Commands

Logger address space resources and LOGR CDS info:

D Logger,L [,options] LOGR CDS logstream view options
D Logger,STR [,options] LOGR CDS structure view options
D Logger,C [,options] Logger address space log stream connection

SETLOGR FORCE[,options] take (force) action on identified resources

Logger CTrace:

TRACE CT,ON,COMP=SYSLOGR,PARM=CTILOG00#,ON,OPTIONS('ALL')
or see CTILOG00 parmlib member

Operations /Commands

Structure related commands (XES):

D XCF,STR,STRNAME=strname
D CF

SETXCF START,REBUILD,STRNAME=strname
SETXCF START,REBUILD,DUPLEX,STRNAME=strname

SETXCF STOP,REBUILD,DUPLEX,STRNAME=strname,KEEP=NEW | OLD
SETXCF FORCE,CON,STRNAME=strname,CONNAME=ALL
SETXCF FORCE,STRUCTURE,STRNAME=strname

Operations /Commands

Serialization / contention related commands:

D GRS,C
D GRS,RES=(SYSZLOGR,*)

DFHSM migrate/recall related commands:

F DFHSM,QUERY
F DFHSM,CANCEL
F DFHMS,SETSYS

Reference: DFSMShsm Storage Administration Guide

Operations /Commands

Operlog related commands:

D C or D C,HC

VARY OPERLOG,HARDCPY
VARY OPERLOG,HARDCPY,OFF

Logrec related commands:

D LOGREC

SETLOGRC LOGSTREAM
SETLOGRC DATASET

Common Setup Issues

Common setup issues

- ✓ LOGR CDS not formatted at highest level
- ✓ Incorrect VSAM ShareOptions (datasets not accessible on all systems)
- ✓ Logger not allocating data sets with expected sizes or to proper volumes
- ✓ CF structure size not appropriate for log stream use

Common Setup Issues

☑ LOGR CDS not formatted at highest level

- D XCF,COUPLE,TYPE=LOGR shows format level
- take steps to get HBB7705 format level LOGR CDS as primary

☑ Incorrect VSAM ShareOptions (datasets not accessible on all systems)

- Logger requires datasets associated with a Coupling Facility type logstream in a PLEXCFG=MULTISYSTEM environment to have VSAM SHROPTS of (3,3)
- allows proper sharing for systems in sysplex
- automatically updated in z/OS v1r12

Common Setup Issues

☑ Logger not allocating data sets with expected sizes or to proper volumes

- Check on DFSMS classes and log stream attributes:

LS_DATACLAS	STG_DATACLAS
LS_STORCLAS	STG_STORCLAS
LS_MGMTCLAS	STG_MGMTCLAS
LS_SIZE	STG_SIZE

Common Setup Issues

- ☑ CF structure size not appropriate for log stream use

  - current size can be determined via command:

    D XCF,STR,STRNAME=str_name

CFSizer

- Web-based tool that is the IBM recommended method to calculate structure sizes

- http://www-1.ibm.com/servers/eserver/zseries/cfsizer/

Use reasonable LOGSNUM value in the LOGR structure definition

Appendices / additional references

- D: z/OS System Logger Parallel Sysplex Infrastructure Summary
- E: Good Reading References
- F: Sample JCL & Sample Outputs

Appendix D: z/OS System Logger Parallel Sysplex Infrastructure Summary

- Disclaimer:
  - Following tables are intended to show a summary view of the maximum limits/numbers associated with z/OS log stream exploitation in a single sysplex.
  - No explicit nor implied commitments are made concerning any future support or updates to this area.

Appendix D: z/OS System Logger Parallel Sysplex Infrastructure Summary

Logger Constraint	Description	Current Limit
LOGR couple data set	Contains system logger policy and log stream resource state information on a sysplex basis.	Only one active primary LOGR CDS in sysplex. One alternate also allowed (copy of primary in ready state in case of PSWITCH)
Number LSR records	System logger records in LOGR CDS that determine overall number of log streams allowed to be defined for the sysplex.	32,767 These record limits (and the 2 below) are defined in z/OS MVS Setting Up a Sysplex, chapter 10, IXCL1DSU utility for LOGR policy.
Number LSTRR records	System logger records in LOGR CDS that determine overall number of CF structures to be used by logger.	32767
Number DEXTENT records	System logger records in LOGR CDS that determine overall number of log stream data set directory extents (allows for 168 offload data sets per DSEXTENT).	99998

Appendix D: z/OS System Logger Parallel Sysplex Infrastructure Summary

Logger Constraint	Description	Current Limit
Number of systems connected to a log stream	z/OS images that can concurrently connect to a single log stream (based on type).	Up to 32 systems concurrently for CF structure based log stream given # systems in the sysplex. Only 1 system at a time for DasdOnly log stream.
Number of log streams in a single CF structure	Log streams mapped (that can be defined) in the same CF structure resource.	512, but recommended to limit 10 or less since can have significant impact of CF structure resource use.
Number connected CF structures for logger	Logger Production CF structures that can be connected concurrently.	255 on one z/OS image. 1024 for sysplex assuming CF allocated limit.
Number connected log streams in CF structures	CF structure based production log streams that can be connected concurrently.	32,767 on one z/OS image given LSR limit, but storage usage likely makes this much smaller. 32,767 for sysplex given LSR limit.

Appendix D: z/OS System Logger Parallel Sysplex Infrastructure Summary

Logger Constraint	Description	Current Limit
Number connected CF structures for logger – GROUP(TEST)	Logger TEST CF structures that can be connected concurrently.	63 on one z/OS image. 1024 for sysplex assuming CF allocated limit.
Number connected log streams in CF structures – GROUP(TEST)	CF structure based test log streams that can be connected concurrently.	32,256 on one z/OS image, but storage usage likely makes this much smaller. 32,767 for sysplex given LSR limit.
Number connected DasdOnly log streams	DasdOnly production log streams that can be connected concurrently.	16,384 on one z/OS image, but storage usage likely makes this much smaller. 32,767 for sysplex given LSR limit.
Number connected DasdOnly log streams – GROUP(TEST)	DasdOnly test log streams that can be connected concurrently.	4096 on one z/OS image. 32,767 for sysplex given LSR limit.
Number of connectors to a single log stream – any type	Concurrent connection instances from exploiters that connect to & share (either read or write) data in a single log stream	No specific limit, but storage usage will govern allowance per system.

Appendix D: Logger Parallel Sysplex Infrastructure Summary (cont.)

Logger Constraint	Description	Current Limit
Size of a “log block” object in a log stream	Maximum Log block size in a log stream, which is the data unit that can be atomically read/written by Logger operations.	64K-4 bytes (65,532)
Staging data set size	Maximum size of an individual log stream staging data set.	Up to 4 GB
Offload data set size	Maximum size of an individual log stream offload data set.	Up to 4 GB
Number of offload data sets	DASD offload data sets for a single log stream.	168 instances, plus 168 more per available DSEXTENT record.

Appendix E: Good Reading – Setup (and planning)

- z/OS MVS Setting Up a Sysplex
  - Chapter 9. Planning for system logger applications
  - LOGR parameters for format utility
  - LOGR Parameters for Administrative Data Utility
  - Calculating the Size of List Structures for System Logger Applications

- System Programmer's Guide to: z/OS System Logger
  - IBM Redbooks Website:
  - ww.redbooks.ibm.com/redbooks/pdfs/sg246898.pdf

Good Reading – Setup (and planning)

- z/OS MVS Initialization and Tuning Reference
  - statements/parameters for IEASYSxx
  - Parmlib members (COUPLExx, GRSRNLxx, IEFSSNxx)
  - Calculating the Size of List Structures for System Logger Applications

- z/OS MVS Installation Exits
  - IXGSEXIT – log stream subsystem default exit

- z/OS Planning Global Resource Serialization

- z/OS Planning Operations

Good Reading – Programming Resources

- z/OS MVS Assembler Services Reference IAR-XCT
  - IXGBRWSE – Browse/Read a Log Stream
  - IXGCONN – Connect/Disconnect to Log Stream
  - IXGDELET – Deleting Log Data from a Log Stream
  - IXGIMPRT – Import Log Blocks
  - IXGINVNT – Managing the LOGR Inventory Couple Data Set
  - IXGOFFLD – Initiate Offload to DASD Log Data Sets
  - IXGQUERY – Query a Log Stream for Information
  - IXGUPDAT – Update Log Stream Control Information
  - IXGWRITE – Write Log Data to a Log Stream

- z/OS MVS Authorized Assembler Services Reference EDT-IXG
  - For above services that have authorized environments and/or parameter requirements

Good Reading – Programming Resources

- z/OS MVS Assembler Services Guide
  - chapter 27 ‘Using System Logger Services’

- z/OS MVS Authorized Assembler Services Guide
  - chapter 30 ‘Using System Logger Services’
  - also covers ENF48 signaling

- z/OS Data Areas, Vol 3(IVT-RCWK)
  - IXG- mapping macros

Good Reading – Operations

- z/OS MVS System Commands
  - DISPLAY Commands
  - SETLOGR Command
  - FORCE IXGLOGR,ARM (to restrart IXGLOGR @space)
  - Start IXGLOGRS (to restrart IXGLOGR @space)

- z/OS MVS System Management Facilities (SMF)
  - Setting up SMF logging
  - System Logger accounting
  - Record Type 88 (58) – System Logger Data

Good Reading – Operations

- z/OS Health Checker Users Guide
  - System logger checks (IBMIXGLOGR)
  - Define log streams to keep a record of the check output

- z/OS MVS System Messages vol 10 (IXC-IZP)
  - IXG messages
  - IXGH messages

Good Reading – Problem Determination

- z/OS MVS Diagnosis: Reference
  - Chapter 24 System Logger
    (includes logger sys1.samplib JCL utilities)

- z/OS MVS Diagnosis: Tools and Service Aids
  - SYSLOGR component trace
  - Obtaining records from the logrec log stream
  - Log Stream Subsystem Data Set JCL Specification

- IPCS Commands
  - LOGGER Subcommand – System Logger @Space Data
  - Appendix D. Control Blocks and Data Areas Scanned, Mapped &Formatted

- System Codes
  - 1C5
  - System Completion Code to Module Table

Appendix F: Sample JCL & Sample Outputs

- IFCEREP1
- Define structure and logstream
- Update logstream definition
  - RETPD / AUTODELETE
  - other keywords
- Delete logstream definition
  - No connection to the logstream can exist to allow delete to occur
- Delete structure definition from the LOGR CDS
  - Note: Logstreams cannot be defined to structure when trying to delete it
- IXCMIAPU Detailed Log Stream List Output

SAMPLE EREP JCL:

```
//EREP     JOB  MSGLEVEL=(1,1),NOTIFY=IBMUSER,MSGCLASS=A
//************************START OF SPECIFICATIONS*********************
//* *-------------------------------------------------------------*/  *
//* * Read Records from a Log Stream via LOGR Subsystem Interface */  *
//* *-------------------------------------------------------------*/  *
//*  'HIST' is a necessary parm to make IFCEREP1 use Logstream.       *
//*  'ACC=N' says not to place output to the ACCDEV DD  card.         *
//*  'ACC=Y' would mean that you have to specify ACCDEV DD  DSN=.     *
//*************************END OF SPECIFICATIONS**********************
//STEP1    EXEC PGM=IFCEREP1,REGION=4M,
//    PARM=('PRINT=PS',
//          'ACC=N',
//          'HIST',
//          'TYPE=ACDEHIMOSX')
//*
//ACCIN    DD   DSN=SYSPLEX.LOGREC.ALLRECS,
//    DCB=(RECFM=VB,BLKSIZE=4000),
//    SUBSYS=(LOGR,IFBSEXIT,
//*           ,'DEVICESTATS')
//*           'FROM=OLDEST,TO=YOUNGEST')
//*           'TO=(2001/090)')
//*           'FROM=OLDEST,TO=YOUNGEST',DELETE)
//*           'FROM=OLDEST,TO=(2000/062)',DELETE)
//*           'FROM=(2000/150),TO=YOUNGEST')
//            'FROM=(2001/060,07:00),TO=(2001/090,23:59),GMT')
//*           EREP will NOT allow a FROM=time and ,DELETE
//*
//DIRECTWK DD   UNIT=SYSDA,SPACE=(CYL,5,,CONTIG)
//TOURIST  DD   SYSOUT=A,DCB=BLKSIZE=133
//EREPPT   DD   SYSOUT=A,DCB=BLKSIZE=133
//SYSABEND DD   SYSOUT=A
//ACCDEV   DD   DUMMY
//SYSIN    DD   DUMMY
//SERLOG   DD   DUMMY
/*
```

Define Structure and Logstream JCL

```
//DEFSTRM  JOB  MSGLEVEL=(1,1),NOTIFY=IBMUSER,MSGCLASS=A
//STEP1    EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
   DATA TYPE(LOGR) REPORT(YES)
   DEFINE STRUCTURE
          NAME(LIST13)
          LOGSNUM(15)
   DEFINE LOGSTREAM
          NAME(SYSPLEX.OPERLOG)
          STRUCTNAME(LIST13)
          STG_DUPLEX(NO)
          HLQ(DONNYC)
          LOWOFFLOAD(0)
          HIGHOFFLOAD(80)
          STG_SIZE(20)
          LS_SIZE(20)
/*
```

Define Structure and Logstream Output (partial)

```
LOGR COUPLE DATA SET FORMAT LEVEL: HBB7705

 /* Functional Items:  */
 /* SMDUPLEX(1)   */

Type                              Formatted              In-use
-------------------               ---------              ------
LSR (Log Stream)                        25                   3
LSTRR (Structure)                       25                   5
DSEXTENT (Data Set Extent)              15                   0

LOGSTREAM NAME(SYSPLEX.OPERLOG) STRUCTNAME(LIST13) LS_DATACLAS()
          LS_MGMTCLAS() LS_STORCLAS() HLQ(DONNYC) MODEL(NO) LS_SIZE(20)
          STG_MGMTCLAS() STG_STORCLAS() STG_DATACLAS() STG_SIZE(20)
          LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO) DUPLEXMODE()
          RMNAME() DESCRIPTION() RETPD(0) AUTODELETE(NO) OFFLOADRECALL(YES)
          DASDONLY(NO) DIAG(NO) LOGGERDUPLEX(UNCOND) EHLQ(NO_EHLQ)
GROUP(PRODUCTION)

STRUCTURE NAME(LIST13) LOGSNUM(15)
          MAXBUFSIZE(65532) AVGBUFSIZE(32766)
          EFFECTIVE AVERAGE BUFFER SIZE(1024) GROUP(PRODUCTION)

          LOGSTREAMS CURRENTLY DEFINED TO THIS STRUCTURE(1)
```

Update Logstream JCL

```
//UPDATE  JOB  MSGLEVEL=(1,1),NOTIFY=IBMUSER,MSGCLASS=A
//STEP1    EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DATA TYPE(LOGR) REPORT(YES)
    UPDATE LOGSTREAM
           NAME(SYSPLEX.OPERLOG)
           HIGHOFFLOAD(70)
           LOWOFFLOAD(0)
/*
```

NOTE: Some keywords cannot be updated dynamically (when active connections to the logstream exist). The list of keywords that can be updated dynamically are:

RETPD, AUTODELETE, LS_SIZE, LS_DATACLAS, LS_MGMTCLAS, LS_STORCLAS, OFFLOADRECALL, LOWOFFLOAD, HIGHOFFLOAD, STG_SIZE, STG_DATACLAS, STG_MGMTCLAS, STG_STORCLAS, STG_DUPLEX, DUPLEXMODE, LOGGERDUPLEX, MAXBUFSIZE

Delete Logstream JCL

```
//DELSTRM  JOB  MSGLEVEL=(1,1),NOTIFY=IBMUSER,MSGCLASS=A
//STEP1    EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DATA TYPE(LOGR) REPORT(YES)
    DELETE LOGSTREAM NAME(SYSPLEX.OPERLOG)
/*
```

NOTE: Logger will only process the delete request if there are no active or outstanding connections to the logstream. Issue Display commands first to verify if there are any active connections.

Delete Structure JCL

```
//DELSTRC  JOB  MSGLEVEL=(1,1),NOTIFY=IBMUSER,MSGCLASS=A
//STEP1    EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DATA TYPE(LOGR) REPORT(YES)
    DELETE STRUCTURE NAME(LIST13)
/*
```

NOTE: Logger will NOT process the delete structure request if there are logstreams defined to the structure. These logstreams will first have to be deleted before attempting to delete the structure.

List Logstream JCL

```
//LISTUTIL  JOB  MSGLEVEL=(1,1),NOTIFY=IBMUSER,MSGCLASS=A
//STEP1    EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DATA TYPE(LOGR) REPORT(YES)
    LIST LOGSTREAM NAME(SYSPLEX.OPERLOG) DETAIL(YES)
/*
```

NOTE: Using an * for the NAME field will result in a detail report for every logstream.

List Logstream Output (start)

```
ADMINISTRATIVE DATA UTILITY:  INPUT                          DATA TYPE = LOGR
_____________________________________________________________________________

LINE #     CONTROL CARDS

     1        DATA TYPE(LOGR) REPORT(YES)
     2        LIST LOGSTREAM NAME(SYSPLEX.OPERLOG) DETAIL(YES)
ADMINISTRATIVE DATA UTILITY:  MESSAGES                       DATA TYPE = LOGR
_____________________________________________________________________________



IXG005I LOGR POLICY PROCESSING LINE# 2

  LOGSTREAM NAME(SYSPLEX.OPERLOG) STRUCTNAME(LIST13) LS_DATACLAS()
            LS_MGMTCLAS() LS_STORCLAS() HLQ(DONNYC) MODEL(NO) LS_SIZE(0)
            STG_MGMTCLAS() STG_STORCLAS() STG_DATACLAS() STG_SIZE(0)
            LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO) DUPLEXMODE()
            RMNAME() DESCRIPTION() RETPD(0) AUTODELETE(NO) OFFLOADRECALL(YES)
            DASDONLY(NO) DIAG(NO) LOGGERDUPLEX() EHLQ(NO_EHLQ)

        LOG STREAM ATTRIBUTES:

          User Data:
            00000000000000000000000000000000000000000000000000000000000000000
            00000000000000000000000000000000000000000000000000000000000000000

        LOG STREAM CONNECTION INFO:

          SYSTEMS CONNECTED: 1

          SYSTEM    STRUCTURE         CON CONNECTION  CONNECTION
          NAME      VERSION           ID  VERSION     STATE
          --------  ----------------  --  ----------  ----------
          SY2       C224310813FDE733  01  00010001    Active
```

```
        LOG STREAM DATA SET INFO:

          STAGING DATA SET NAMES: DONNYC.SYSPLEX.OPERLOG.<SUFFIX>

           NUMBER OF STAGING DATA SETS: 0

          DATA SET NAMES IN USE: DONNYC.SYSPLEX.OPERLOG.<SEQ#>

          Ext.   <SEQ#>    Lowest Blockid    Highest GMT        Highest Local
          -----  --------  ----------------  -----------------  --------------
         *00001  A0000000  0000000000000000

          NUMBER OF DATA SETS IN LOG STREAM: 1


        POSSIBLE ORPHANED LOG STREAM DATA SETS:

           NUMBER OF POSSIBLE ORPHANED LOG STREAM DATA SETS: 0

LOGR Inventory Record Summary:

LOGR COUPLE DATA SET FORMAT LEVEL: HBB7705
ADMINISTRATIVE DATA UTILITY:  REPORT

 /* Functional Items:  */
 /* SMDUPLEX(1)   */

Type                            Formatted             In-use
-------------------             ---------             ------

LSR (Log Stream)                      25                  3

LSTRR (Structure)                     25                  5

DSEXTENT (Data Set Extent)            15                  0
```

```
LOGSTREAM NAME(SYSPLEX.AUTODEL) STRUCTNAME(LIST15) LS_DATACLAS()
          LS_MGMTCLAS() LS_STORCLAS() HLQ(DONNYC) MODEL(NO) LS_SIZE(0)
          STG_MGMTCLAS() STG_STORCLAS() STG_DATACLAS() STG_SIZE(0)
          LOWOFFLOAD(30) HIGHOFFLOAD(50) STG_DUPLEX(NO) DUPLEXMODE()
          RMNAME() DESCRIPTION() RETPD(0) AUTODELETE(NO) OFFLOADRECALL(YES)
          DASDONLY(NO) DIAG(NO) LOGGERDUPLEX(UNCOND) EHLQ(NO_EHLQ) GROUP(PRODUCTION)

LOGSTREAM NAME(SYSPLEX.LOGREC.ALLRECS) STRUCTNAME(LIST14) LS_DATACLAS()
          LS_MGMTCLAS() LS_STORCLAS() HLQ(DONNYC) MODEL(NO) LS_SIZE(0)
          STG_MGMTCLAS() STG_STORCLAS() STG_DATACLAS() STG_SIZE(0)
          LOWOFFLOAD(65) HIGHOFFLOAD(90) STG_DUPLEX(NO) DUPLEXMODE()
          RMNAME() DESCRIPTION() RETPD(0) AUTODELETE(NO) OFFLOADRECALL(YES)
          DASDONLY(NO) DIAG(NO) LOGGERDUPLEX(UNCOND) EHLQ(NO_EHLQ) GROUP(PRODUCTION)

LOGSTREAM NAME(SYSPLEX.OPERLOG) STRUCTNAME(LIST13) LS_DATACLAS()
          LS_MGMTCLAS() LS_STORCLAS() HLQ(DONNYC) MODEL(NO) LS_SIZE(20)
          STG_MGMTCLAS() STG_STORCLAS() STG_DATACLAS() STG_SIZE(20)
          LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO) DUPLEXMODE()
          RMNAME() DESCRIPTION() RETPD(0) AUTODELETE(NO) OFFLOADRECALL(YES)
          DASDONLY(NO) DIAG(NO) LOGGERDUPLEX(UNCOND) EHLQ(NO_EHLQ) GROUP(PRODUCTION)

STRUCTURE NAME(LIST01) LOGSNUM(10)
          MAXBUFSIZE(65532) AVGBUFSIZE(32766)
          EFFECTIVE AVERAGE BUFFER SIZE(32766) GROUP()

          LOGSTREAMS CURRENTLY DEFINED TO THIS STRUCTURE(0)

STRUCTURE NAME(LIST02) LOGSNUM(10)
          MAXBUFSIZE(65532) AVGBUFSIZE(32766)
          EFFECTIVE AVERAGE BUFFER SIZE(32766) GROUP()

          LOGSTREAMS CURRENTLY DEFINED TO THIS STRUCTURE(0)
```

```
STRUCTURE NAME(LIST13) LOGSNUM(15)
          MAXBUFSIZE(65532) AVGBUFSIZE(32766)
          EFFECTIVE AVERAGE BUFFER SIZE(1024) GROUP(PRODUCTION)
          LOGSTREAMS CURRENTLY DEFINED TO THIS STRUCTURE(1)

STRUCTURE NAME(LIST14) LOGSNUM(5)
          MAXBUFSIZE(65532) AVGBUFSIZE(32766)
          EFFECTIVE AVERAGE BUFFER SIZE(32766) GROUP(PRODUCTION)

          LOGSTREAMS CURRENTLY DEFINED TO THIS STRUCTURE(1)

STRUCTURE NAME(LIST15) LOGSNUM(5)
          MAXBUFSIZE(65532) AVGBUFSIZE(32766)
          EFFECTIVE AVERAGE BUFFER SIZE(32766) GROUP(PRODUCTION)

          LOGSTREAMS CURRENTLY DEFINED TO THIS STRUCTURE(1)
```

List Logstream Output (end)

Appendix A – OA36662 – details

- System Logger IXGWRITE behavior is enhanced in OA36662 to allow requests to be attempted after the previous IXGWRITE failed due to return code IxgRetCodeError(8) reason codes IxgRsnCodeCFLogStreamStorFull (860) or IxgRsnCodeWowError (85D). The IXGWRITE attempt will be allowed when the writer is authorized, has not encountered a staging data set full condition, has not encountered a directory full condition, and is not a MODE=ASYNCNORESPONSE IXGWRITE request. If the conditions are not met, the IXGWRITE will be rejected with a full type condition as before.

```
For IxgRsnCodeCFLogStreamStorFull (860) or IxgRsnCodeWowError (85D) on
IXGWRITE requests, while waiting for IxgenfLogstreamsAvailable event with
IxgenfLogstreamStorageAvailable reason, authorized application can also
retry the request after short wait. When the structure becomes full logger
starts offloading, and logger will produce the enf signal when the offload
processing is able to relieve the full condition. For these two log stream
structure full type conditions, Logger allows authorized IXGWRITE requests
(with exception for MODE=ASYNCNORESPONSE type requests) to be attempted as
the structure offloads, and these requests may complete successfully
before the enf signal is received.
```

Appendix B – OA36172 – IXGANSAA changes

- IXGCONN, IXGWRITE and IXGQUERY are enhanced in OA36172 for Logger to return usage information for IXGWRITE requests in the IXGANSAA area. The changes are as follows.
- IXGCONN and IXGWRITE have a new flag in the IXGANSAA that is turned on when the support is allowed. New flag Ansaa_WriteTriggersReturned will be added to the first position of Ansaa_Flags2.

```
ANSAA_WRITETRIGGERSRETURNED EQU X'80' For IXGCONN IXGCONN                *
                            REQUEST=CONNECT AUTH=WRITE requests: When ON, *
                            indicates that log stream primary storage     *
                            consumption info will be returned in          *
                            Ansaa_WriteTriggers for successful IXGWRITE   *
                            requests (RETCODE = 0 or 4). For IXGWRITE     *
                            requests: When ON, indicates that data in     *
                            Ansaa_WriteTriggers has been returned.
```

Appendix B – OA36172 – IXGANSAA changes

- IXGWRITE will have additional fields showing storage consumption turned on in the IXGANSAA field Ansaa_Gaps will be re-defined to return service specific information

```
ANSAA_SERVICESPECIFIC DS 0CL8
         ORG   ANSAA_SERVICESPECIFIC
ANSAA_IXGDELET DS 0CL8   IXGDELET information
ANSAA_GAPS DS  0CL8      Gap information
ANSAA_GAPS_NEXT_BLKID DS CL8 Block id of the first valid youngest      *
                         block
         ORG   ANSAA_SERVICESPECIFIC
ANSAA_IXGWRITE DS 0CL8   IXGWRITE information
ANSAA_WRITETRIGGERS DS 0CL3 Data returned when                         *
                         Ansaa_WriteTriggersReturned is on
ANSAA_STRUCTUSEPERCENT DS X Percent of CF structure element objects in *
                         use -Value rounded down -Value between 0 and  *
                         100 -Value not set for DASDONLY log streams
ANSAA_STAGINGUSEPERCENT DS X Percent of staging data set space in use  *
                         -Value rounded down -Value between 0 and 100  *
                         -Value set for DASDONLY log streams and CF    *
                         log streams that duplex to staging data sets
```

Appendix B – OA36172 – IXGANSAA changes

```
ANSAA_WRITEFLAGS DS 0B   Write specific flags                          *
*       * For CF Structure based log streams: The following flags are *
*       * based on percentage of CF structure element objects in use. *
*       * For DASDonly log streams: The following flags are based on  *
*       * percentage of staging data set space in use.
ANSAA_WRITEABOVEHIGHOFFLOAD EQU X'80' IxgWrite above HighOffload       *
                          percentage for log stream
ANSAA_WRITEABOVELOWCAPACITY EQU X'40'
ANSAA_WRITEELEVATEDCAPACITY EQU X'40' Log stream storage usage is at   *
                          an elevated capacity. This IXGWRITE is above  *
                          the 1/3 point between HighOffload % and 100%  *
                          (full) --> (0.33 of delta). Increased         *
                          IXGWRITE activity can pose a possible risk to *
                          the log stream of experiencing a full         *
                          condition.
ANSAA_WRITEABOVENEARCAPACITY EQU X'20'
ANSAA_WRITEIMMINENTCAPACITY EQU X'20' Log stream storage usage is at   *
                          an imminent capacity. This IXGWRITE is above  *
                          the 2/3 point between HighOffload % and 100%  *
                          (full) --> (0.67 of delta). Future IXGWRITE   *
                          activity pose an impending risk to the log    *
                          stream of experiencing a full condition.
```

Appendix B – OA36172 – IXGQBUF changes

- IXGQUERY will be enhanced to return relevant fields HIGHOFFLOAD and LOWOFFLOAD log stream definition information to the IXGQBUF as follow:
  - A flag is added to QBUF_FLAGS to show the support is enabled. QBUF_LS_Offload_Returned is added after QBUF_Using_Physical_Struct2 in QBUF_FLAGS

```
QBUF_LS_OFFLOAD_RETURNED EQU X'08' Log stream definition HIGHOFFLOAD
*
                         and LOWOFFLOAD are returned in fields
*
                         QBUF_Ls_HighOffload QBUF_Ls_LowOffload
```

  - The fields QBUF_LS_HighOffload and QBUF_LS_LowOffload are added to the 2 reserved bytes between QBUF_Ver2_Other_Flag and QBUF_Str_LogSNum

```
QBUF_LS_HIGHOFFLOAD DS X Logstream HighOffload percentage
QBUF_LS_LOWOFFLOAD DS X  Logstream LowOffload percentage
```

Appendix C - Supersizing the structure - example

Dependent Variables:				
Total number of logger writes (IXGRPT1)	6499383		Logger control space	0%
Largest average buffer size (IXGRPT1)	4249		IMS log data, checkpoint 1 2765 MB	
IXGRPT1 interval (minutes)	60			
MAXBUFSIZE (LOGR Policy)	65532			
Minutes between structure checkpoints	5			
Low/High offload delta (minutes)	2			
Desired reaction time (minutes)	10		IMS log data, checkpoint 2 2765 MB	
Computed Values:				
Element Size	512			
Elements need for average buffer size	9		Safety zone 1106 MB	
Entry Size (entry + element list nodes)	738			
Total bytes (entry + elements + ctl)/buf	5346		2 minutes of data 1106 MB	LOWOFFLOAD = 48%
				HIGHOFFLOAD = 56%
Data Rate (MB/Min, 1MB = 2^20 bytes)	553		Reaction Time Space 6083 MB	
Sizing values, in MB:				
Data structure checkpoint #1	2765			FULLTHRESHOLD = 71%
Data structure checkpoint #2	2765			
Safety zone size (20% of 2 chkpts)	1106			
Low/High offload area size	1106			
Reaction time space (reaction time + 1)	6083			100% (13825) MB
Total structure size:	**13825**	**MB**	(Note: 1 MB = 2^20 bytes)	
Threshold values:				
LOWOFFLOAD	48%			
HIGHOFFLOAD	56%			
FULLTHRESHOLD	71%			