



**E14**

# **IMS Shared Queues Special Considerations**

*Luigi Cetorelli*



Anaheim, California

October 23 - 26, 2000

# Agenda

---

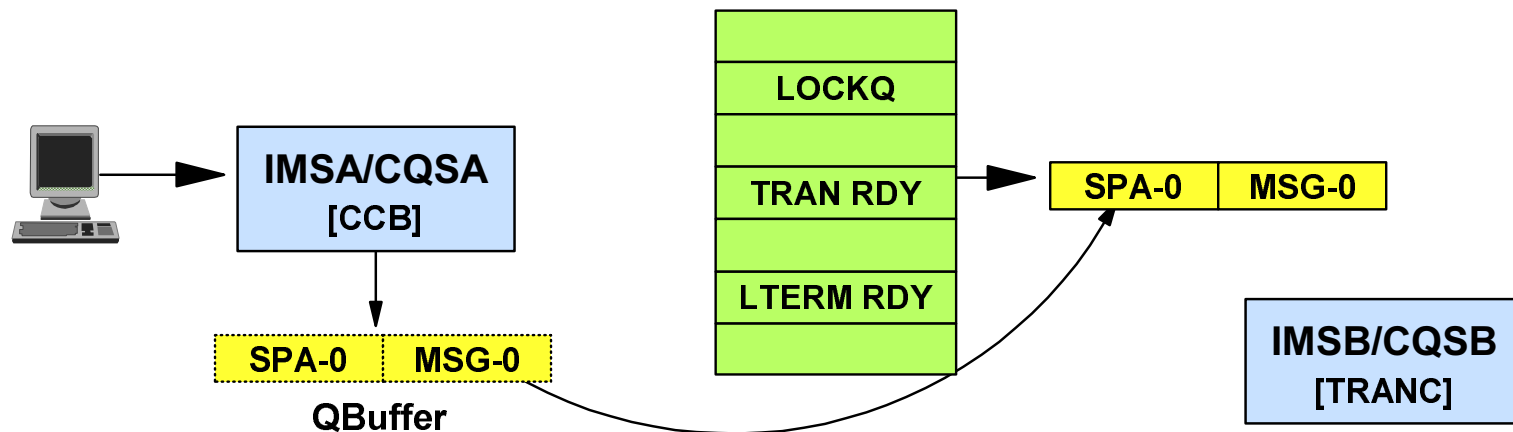
## ▲ Topics

- ▶ Conversational Transactions
- ▶ ETO Multiple Signon
- ▶ ETO Autologon and Shared Printers
- ▶ MSC Considerations
- ▶ Serial Transactions
- ▶ Undefined Resources
- ▶ Security
- ▶ Log Records
- ▶ Miscellaneous
- ▶ Exits

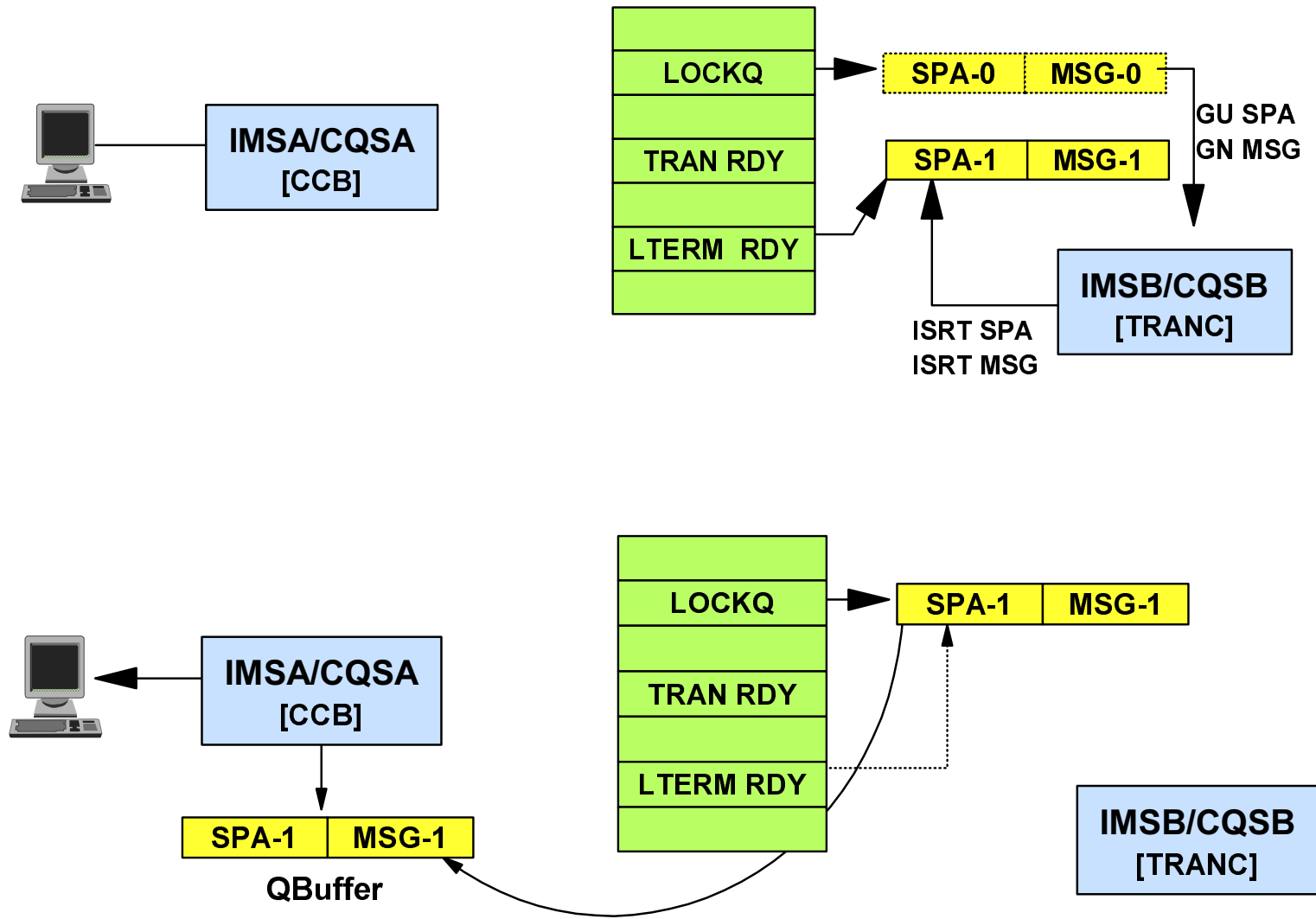
# Conversational Transactions

## ▲ Conversational transactions are supported in Shared Queues

- ▶ Transaction entered on IMSA can be processed on any IMS in the SQ group
- ▶ Front-end IMS puts SPA and Message on the Transaction Ready Queue
  - All IMSs are informed
- ▶ Processing IMS puts SPA and response message on LTERM Ready Queue
  - Front-end IMS delivers message
- ▶ SPA and output message saved in structure on LOCKQ and in F-E IMS QPOOL and in MSGQ structure (LOCKQ)
  - Must be considered when sizing structure and local buffers



# Conversational Transactions ...



# Conversational Transactions ...

▲ **Conversational status is known only to Front-End IMS**

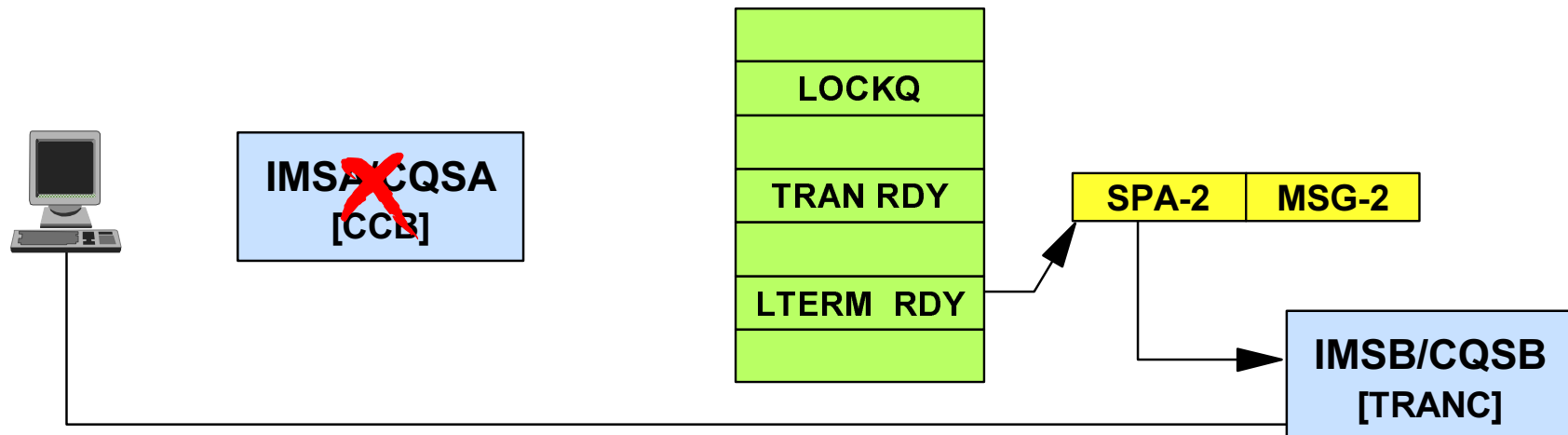
- ▶ Status maintained in F-E control blocks (VTCB, CCB, SPQB)

▲ **If IMSA fails and user logs on to IMSB**

- ▶ IMSB registers interest in LTERM

▲ **If there is a message on the queue when IMSB registers interest**

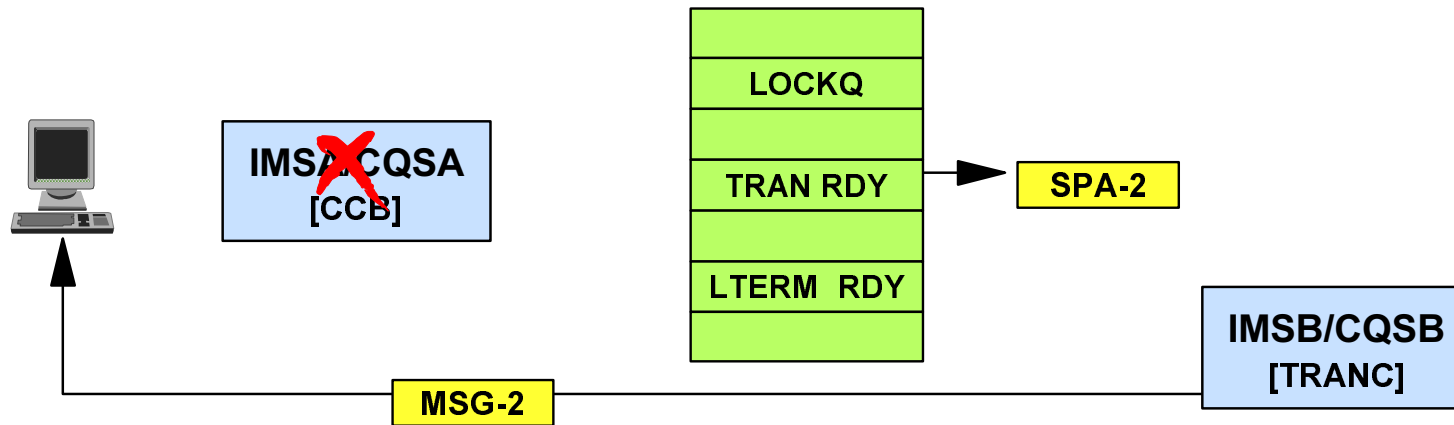
- ▶ Conversational status is unknown to IMSB
- ▶ IMSB will drive conversation abnormal termination exit (DFSCONE0)



# Conversational Transactions ...

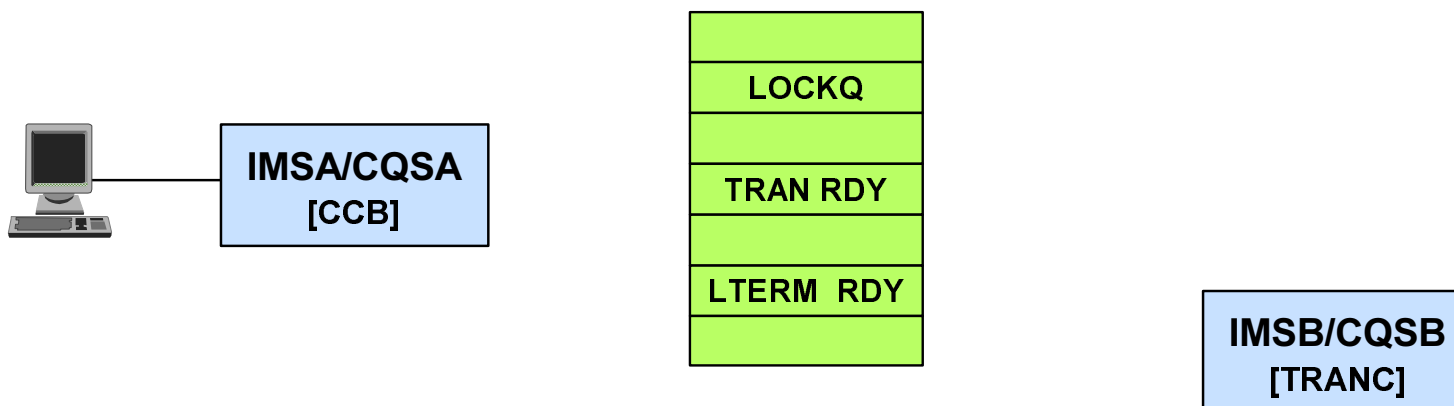
## Exit can

- ▶ Deliver or delete message, requeue or delete SPA



## If user later logs back on to IMSA

- ▶ User is back in conversational mode, last response is gone, terminal hung



# ETO Considerations (Dynamic LTERMs)

## ▲ When user logs on

- ▶ Terminal control blocks (VTCB) are built defining the terminal characteristics

## ▲ When a user signs on

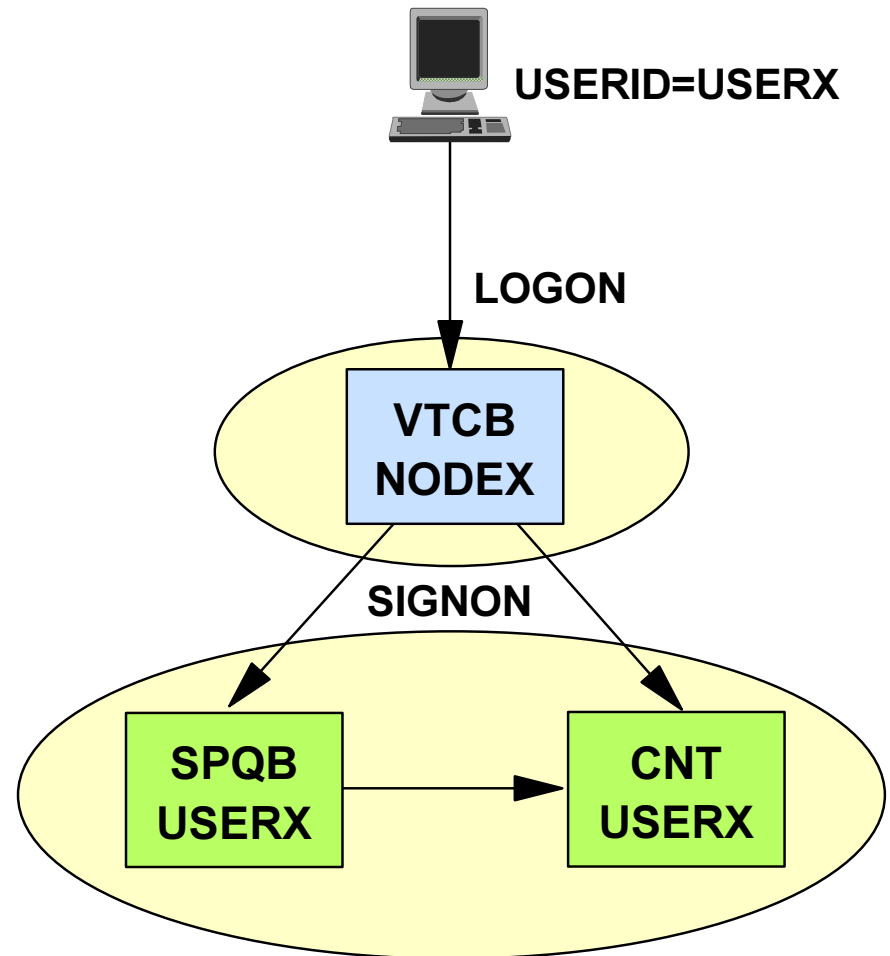
- ▶ User structure is built with USER (SPQB) and LTERM (CNT) control blocks

## ▲ If there is no signon exit

- ▶ LTERM name is set to USERID
  - e.g. USERX

## ▲ If there is a signon exit (DFSSGNX0)

- ▶ Exit can specify LTERM name
  - e.g. USERX or FRED



## ETO Considerations ...

---

- ▲ **Potential problem exists if the same LTERM name is created in multiple IMSs**
  - ▶ Same user signs on to multiple IMSs without Signon Exit
  - ▶ Signon Exit generates the same LTERM name on multiple IMSs
  
- ▲ **When multiple signons are allowed in a single system**
  - ▶ Signon Exit creates unique LTERM names for each signon
    - e.g. USERX1, USERX2, USERX3
  - ▶ Signon Exit can determine whether an LTERM name is already assigned
  
- ▲ **When multiple signons occur in a Shared Queues Group**
  - ▶ Signon Exit does not know what LTERM names have been assigned in other IMSs in the group
  - ▶ May create duplicate LTERM names
    - IMSA: USERX1, USERX2
    - IMSB: USERX1, USERX2
  - ▶ Will cause problems when messages are queued by LTERM name



## ETO Considerations ...

---

### ▲ Possible solution 1

- ▶ Use LTERM-name assignment algorithm that guarantees uniqueness
  - e.g. USERX1A, USERX1B
- ▶ Use a table of valid LTERM names
  - Table is different for each IMS

### ▲ This raises another problem

- ▶ If applications are sensitive to LTERM name
  - e.g. Printer for USERX1 is USERX1P

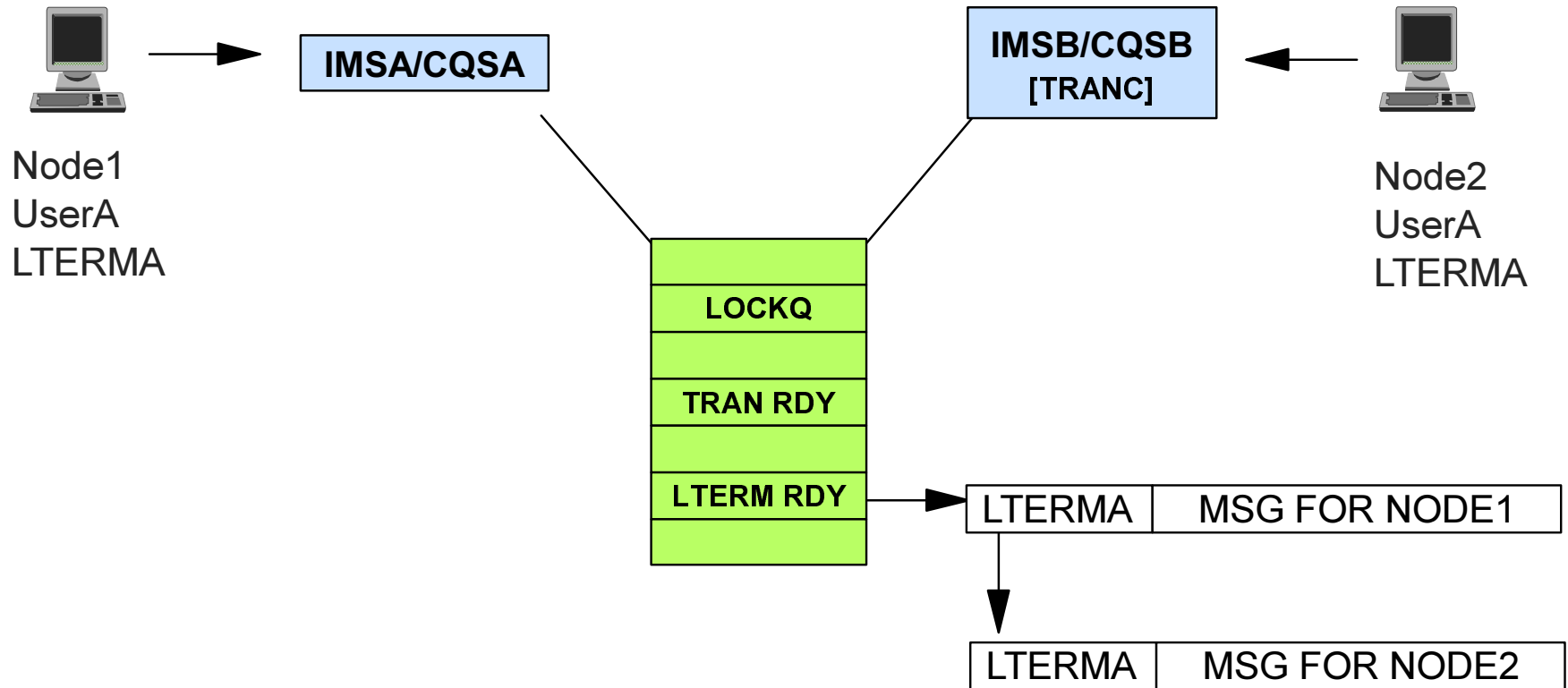
### ▲ Potential solution 2

- ▶ Require user to always sign on to just one IMS
  - LTERM names for multiple signons can be unique

### ▲ But ...

- ▶ Can't prevent user from signing on to multiple IMSs
- ▶ Mitigates some availability benefits

# ETO Considerations ...



Since queuing is by LTERM only, either message could be delivered to either Node1 or Node2.

# Shared Printers and ETO Autologon

---

## ▲ Shared printers and ETO Autologon support

- ▶ Session automatically initiated on system originating output

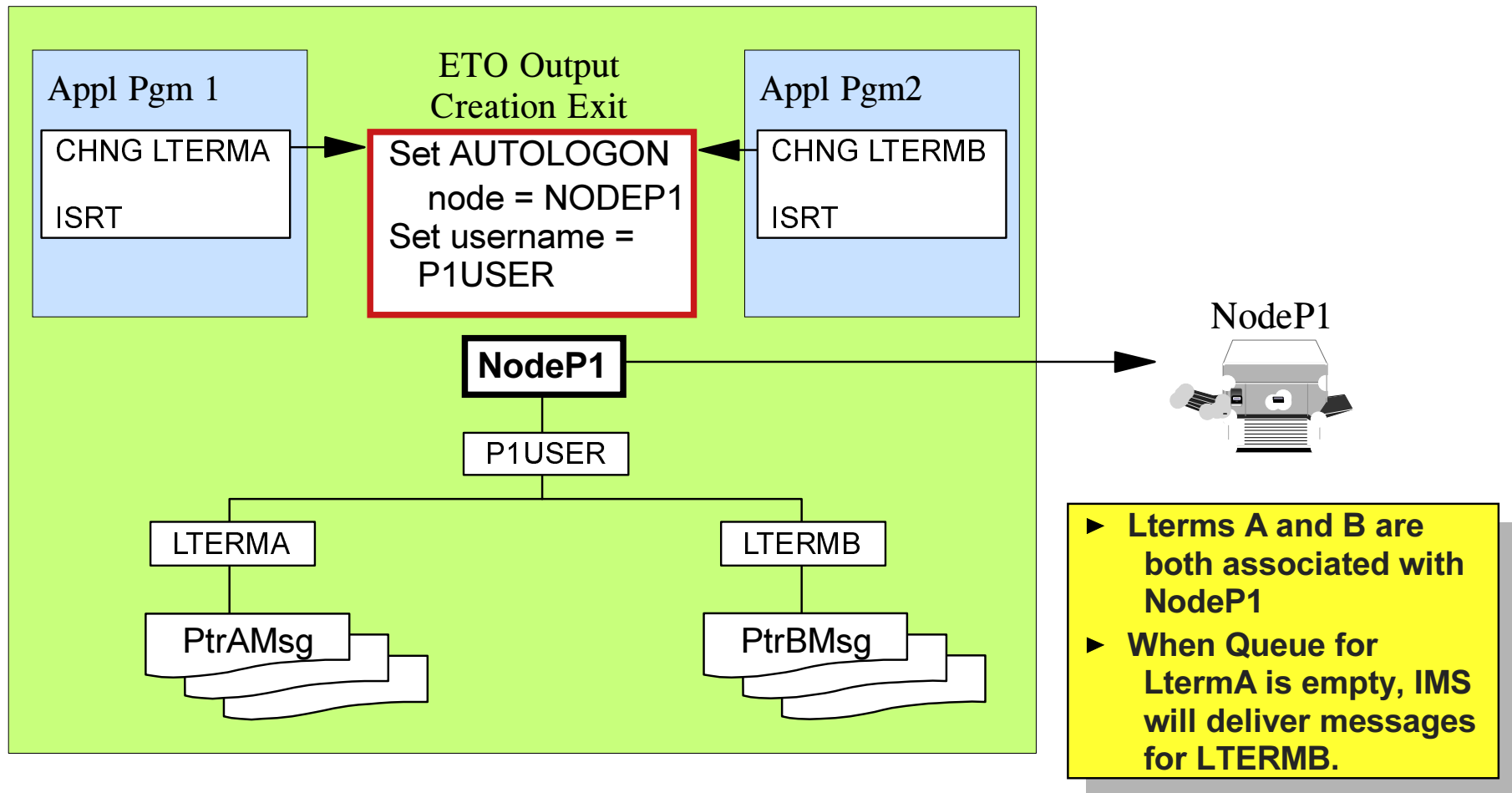
## ▲ Multiple systems may compete to deliver output from a specific queue (LTERM) name

- ▶ Destination is a **Shared Printer** defined in two or more systems
- ▶ Destination is subject to **Autologon** from two or more systems
- ▶ Destination **LTERM is active in two or more systems**

# Autologon with a Single IMS (Reminder)

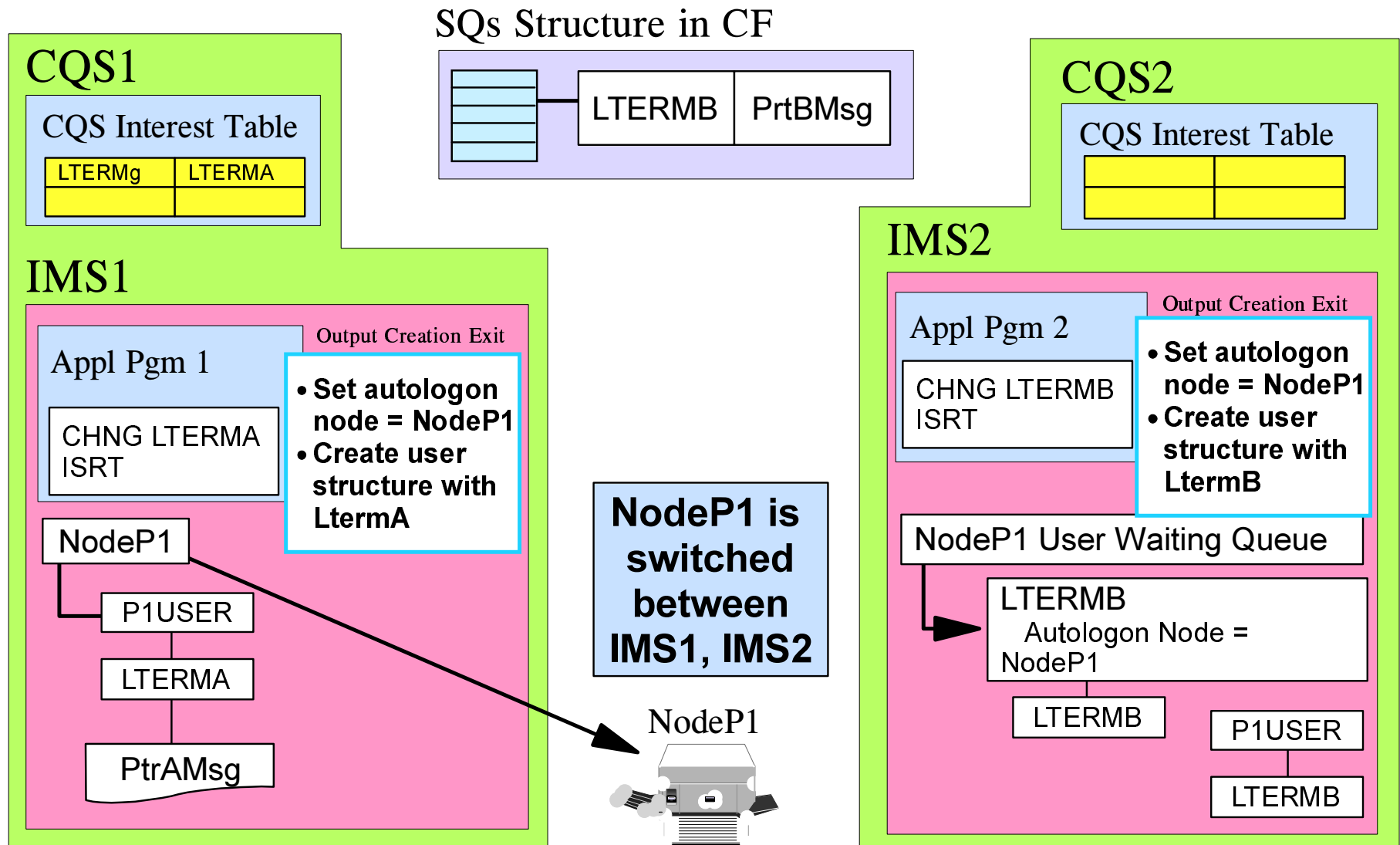
## One Possible Implementation

IMS1



- ▶ Lterms A and B are both associated with NodeP1
- ▶ When Queue for LtermA is empty, IMS will deliver messages for LTERMB.

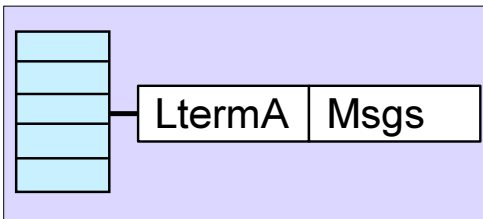
# Autologon With SQs and Multiple IMSs



# Shared Printers - Potential Solution

- Use same Lterm-name on all IMSs
  - ▶ Beneficial use of Synonym Lterms
- Prevent switching by using **NORELRQ**

SQs Structure in CF

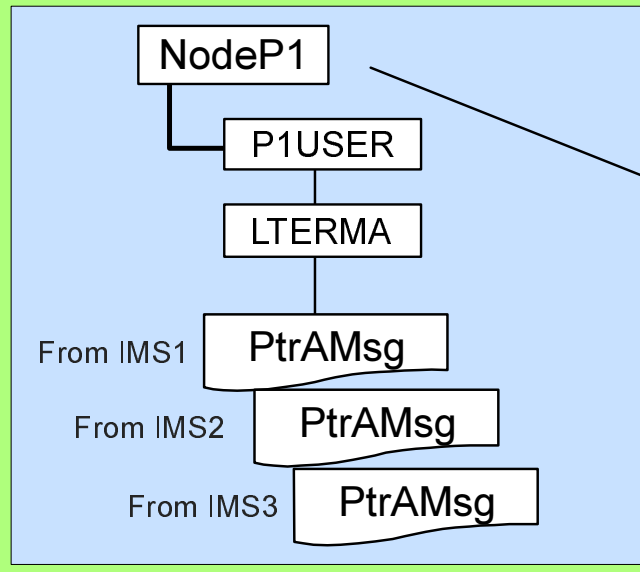


CQS1

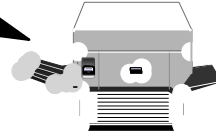
CQS Interest Table

LTERM	LTERMA

IMS1



NodeP1



**All messages delivered by IMS1**

NodeP1 User Waiting Queue

LtermA  
Autologon Node =  
NodeP1

LTERMA

P1USER

LTERMA

IMS2

NodeP1 User Waiting Queue

LtermA  
Autologon Node =  
NodeP1

LTERMA

P1USER

LTERMA

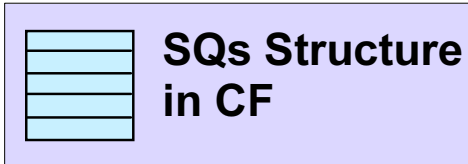
IMS3

# Another Possible Approach

This allows two printers to be used as a single resource. Messages from both IMSs will be intermixed on both printers. It requires a Smart Output Creation exit.

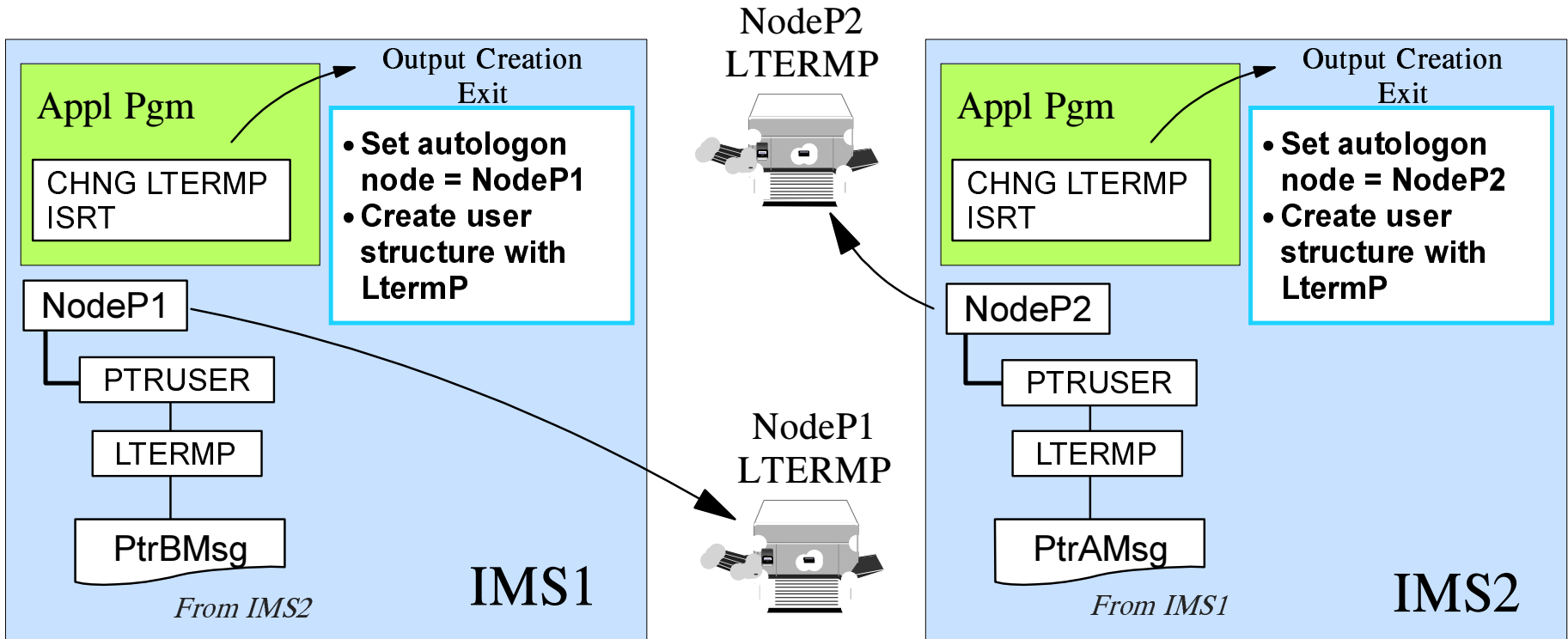
CQS1 Interest Table

LTERM	LTERMP



CQS2 Interest Table

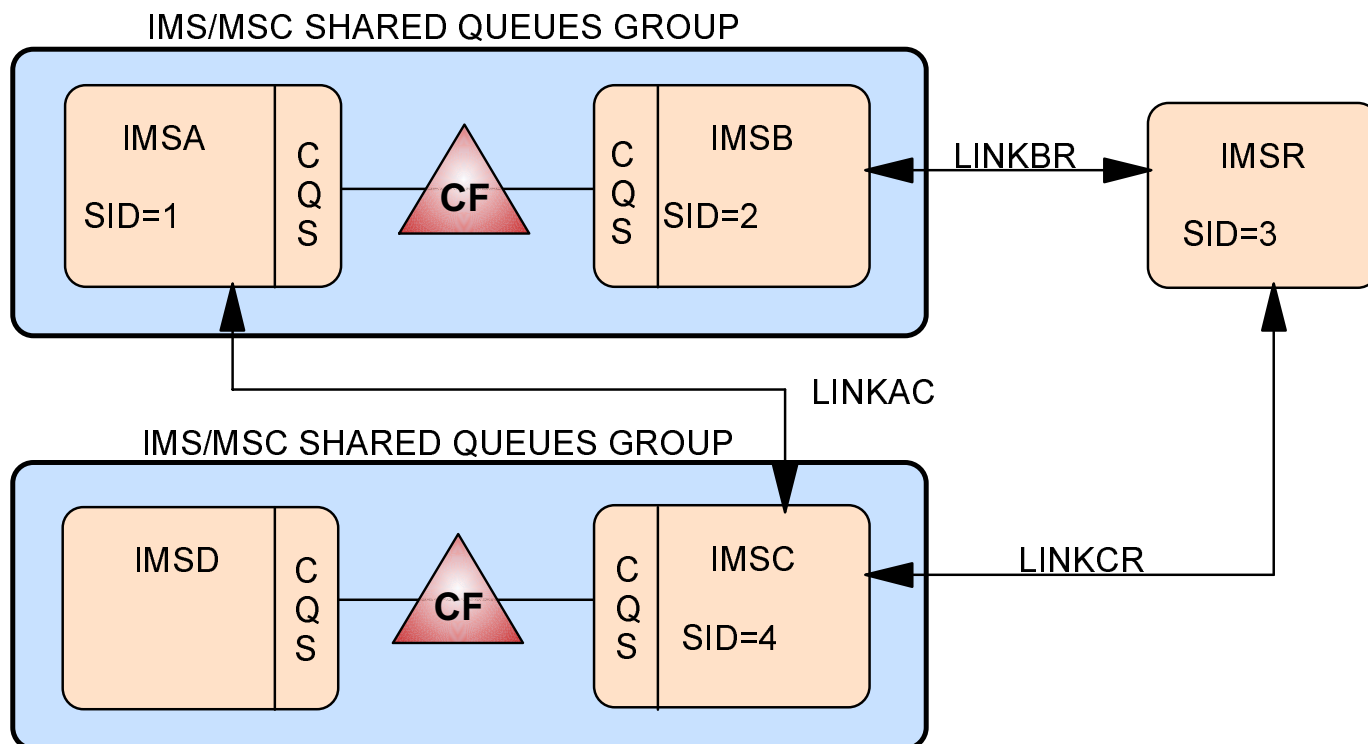
LTERM	LTERMP



## MSC and Shared Queues

### ▲ MSC systems can exist in the Shared Queues environment

- ▶ Messages may be entered to any IMS in a shared queues group and be sent by another IMS in the shared queues group to a remote IMS
- ▶ Messages arriving from a remote IMS may be placed on the shared queue and processed by any IMS with registered interest

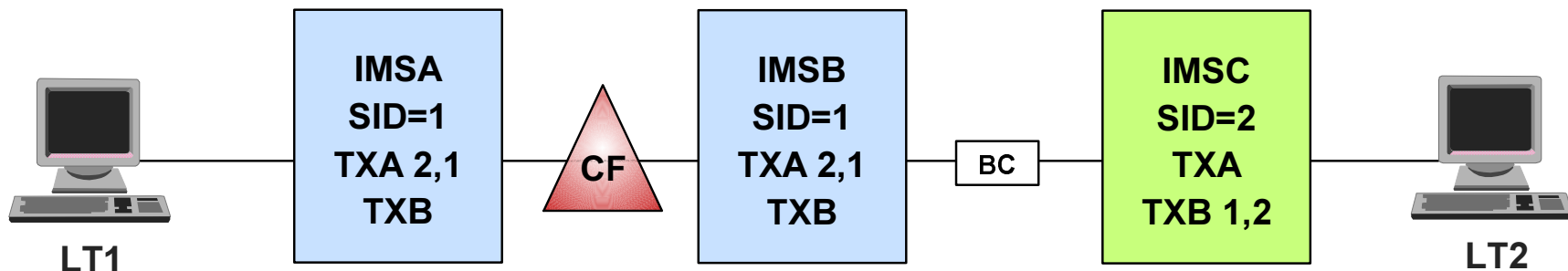




# SYSIDs

## ▲ With MSC in a Shared Queues environment

- ▶ Local SYSIDs are considered to be **local to all IMSs in the Shared Queues Group**
- ▶ Any message destined for any SYSID local to the Shared Queues Group
  - Is placed on the Shared Queues
  - May be retrieved by any IMS in the group with interest in the destination name
- ▶ Enables every IMS in the shared queues group to
  - Process MSC input from any remote IMS (e.g. TXB from LT2 on IMSC)
  - Send MSC output to any remote IMS (e.g. TXA from LT1 to IMSC)



## Cloning MSC System Definitions

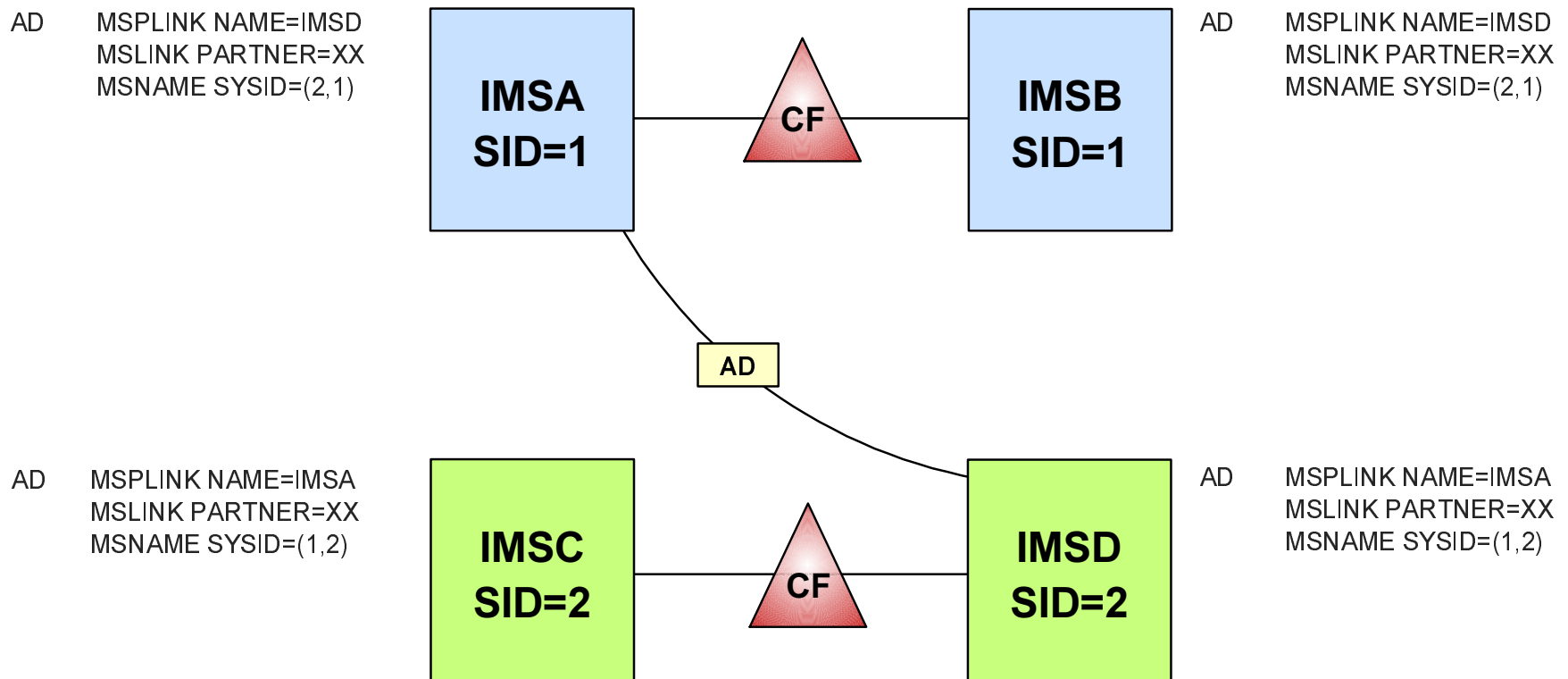
---

### ▲ MSC definitions may be cloned throughout all IMS systems within a shared queues group

- ▶ Only one member of the shared queues group may have a cloned link active with an IMS remote to the shared queues group
  
- ▶ Active links between IMSs within the shared queues group are not allowed (may not be started)
  - DFS2149 PARTNER IMS IN SAME SHARED QUEUES GROUP  
– RESTART ABORTED LINK xxx
  
- ▶ One or more members of the shared queues group can be used to establish back-up connections in the event of link failure

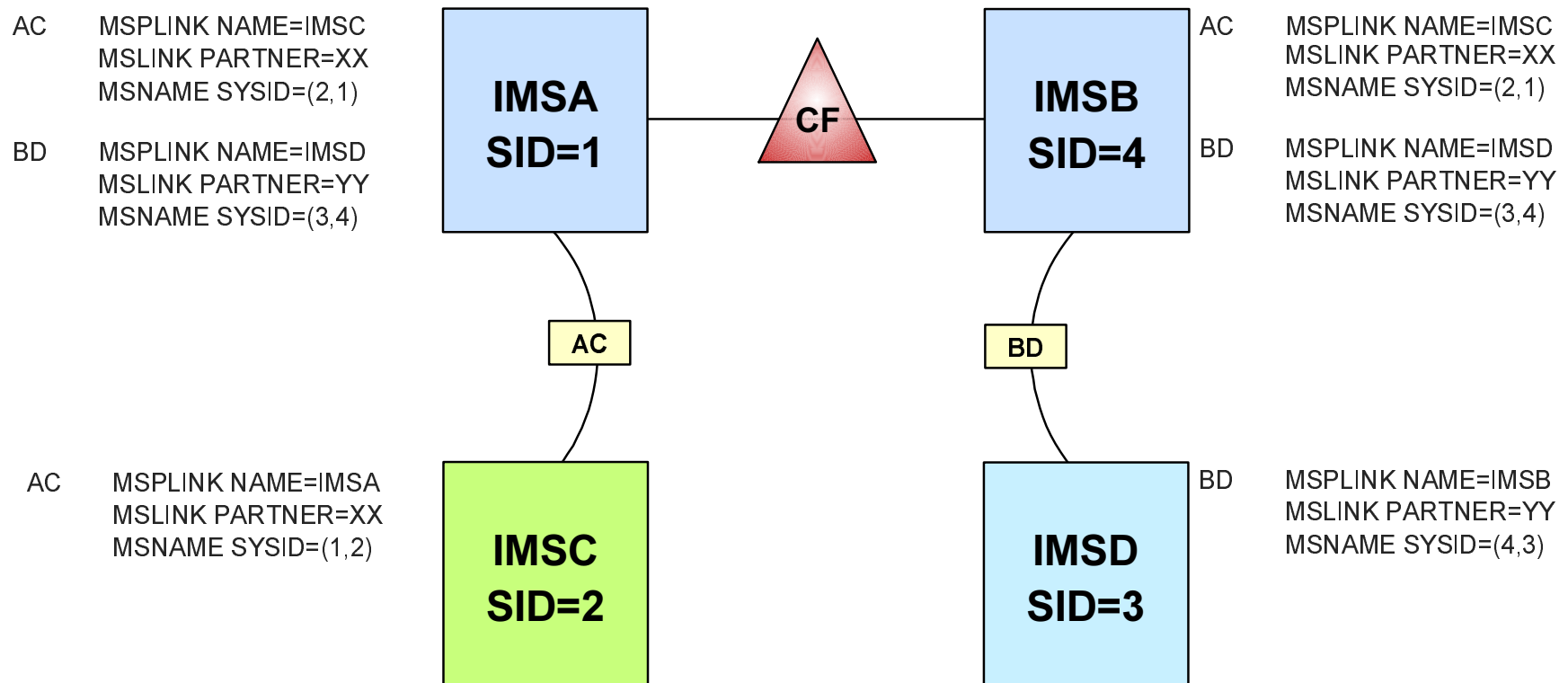
# Cloning MSC Definitions ...

## ▲ Example: Shared Queues Group to Shared Queues Group



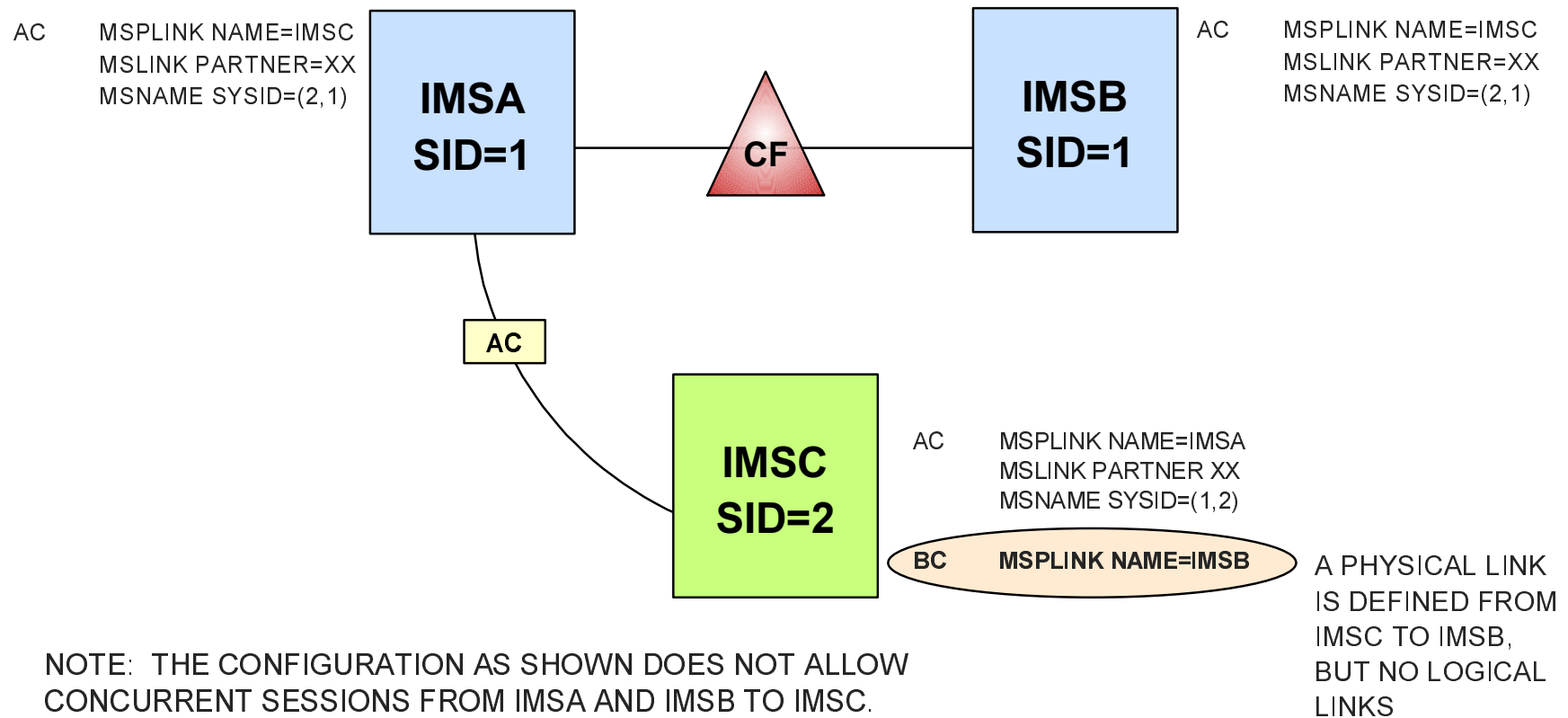
# Cloning MSC Definitions ...

## ▲ Example: Shared Queues Group to multiple remote IMS systems



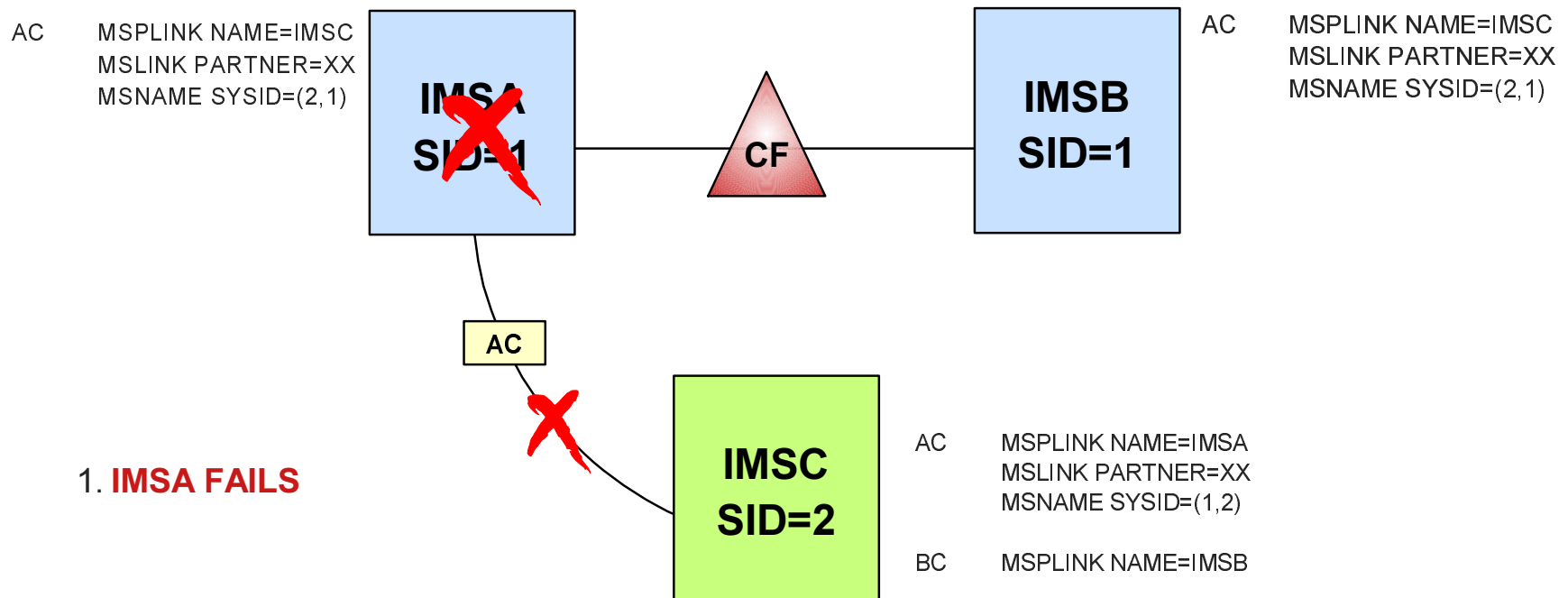
# Cloning MSC Definitions ...

## ▲ Backing up logical links from a shared queues group to a remote IMS



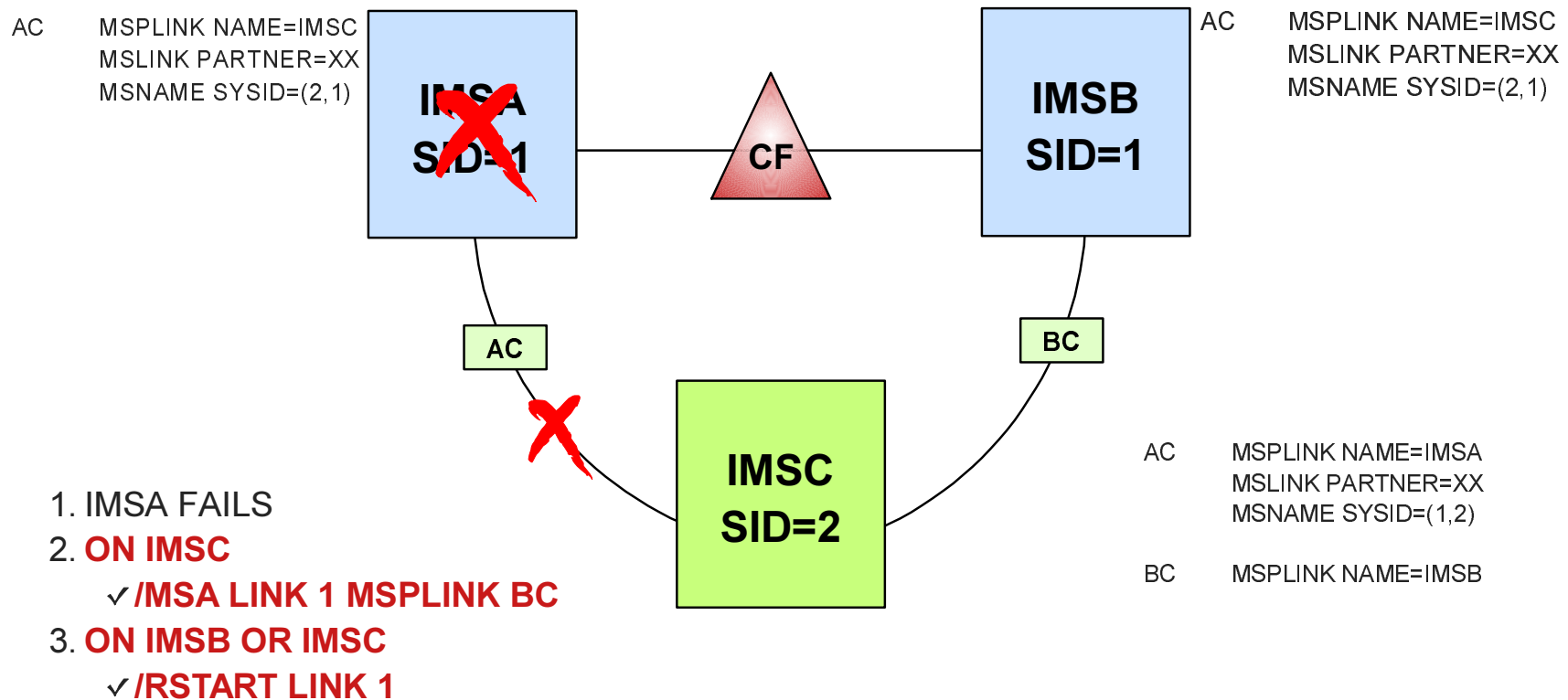
# Cloning MSC Definitions ...

▲ Backing up logical links from a shared queues group to a remote IMS ...



# Cloning MSC Definitions ...

## ▲ Backing up logical links from a shared queues group to a remote IMS ...



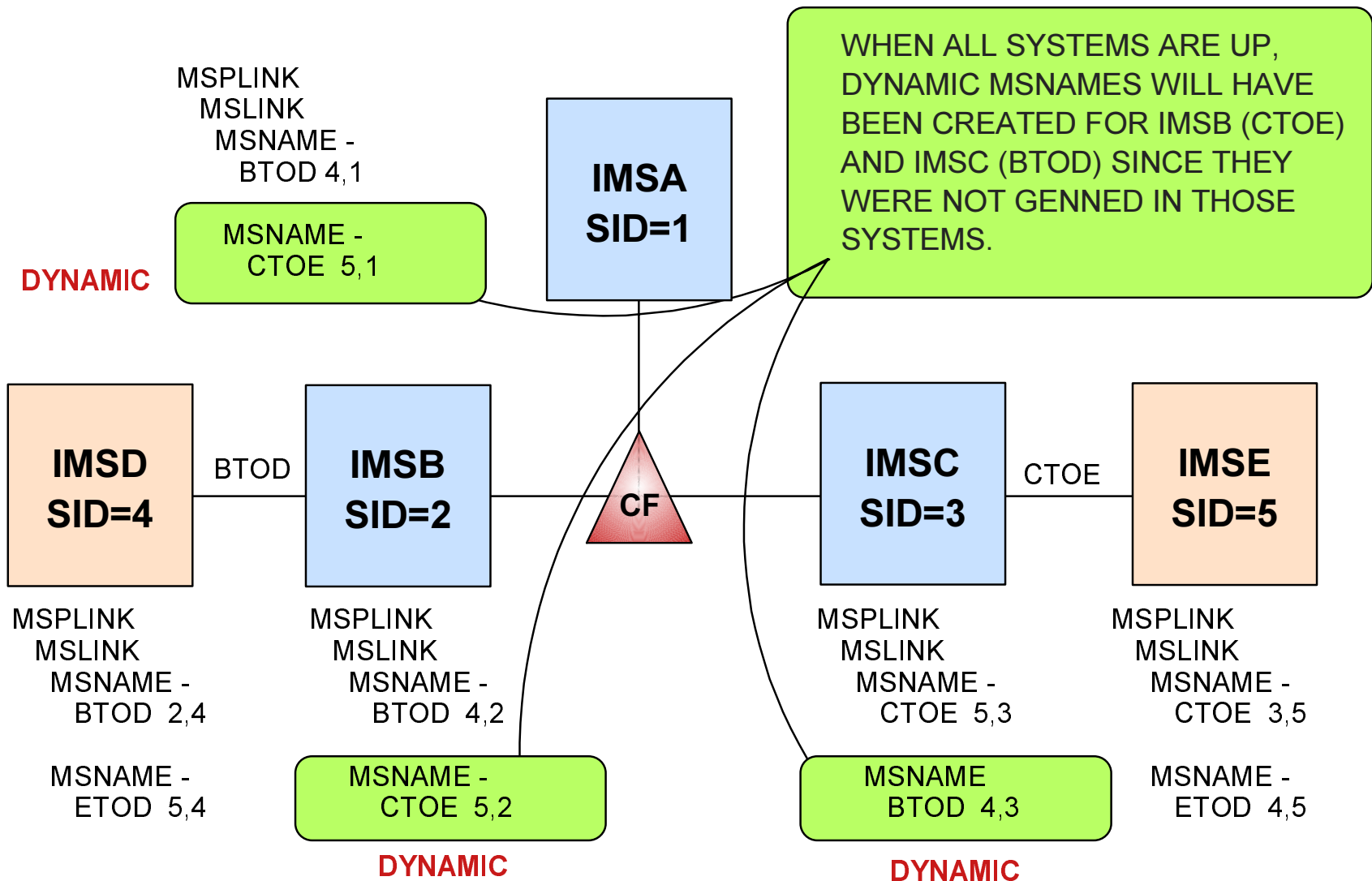
## Dynamic MSNAMES & SYSIDS ...

---

- ▲ For remote **SYSIDs** and their associated **MSNAMES** each **IMS** in the **Shared Queues Group**
  - ▶ May **SYSGEN** the remote **MSNAMES** of **all IMSs in the group**
    - Same **MSNAME** and remote **SYSID**
    - Unique local **SYSID** of this **IMS** or same local **SYSID** of other shared queues group member
    - **If not genned, IMS dynamically creates the remote MSNAMES**
  - ▶ These dynamic **MSNAMES** will never be started
    - Are used to route messages from any intermediate or back-end **IMS** in the group to the real **MSNAME** of the owning **IMS** in the group
    - These **MSNAMES** are known as **dynamic MSNAMES** and are needed to be able to queue messages to the Remote Ready Queue



# Dynamic MSNAME Illustration



# Serial Transactions

---

## ▲ Without shared queues

- ▶ Transactions defined as SERIAL=YES will be executed in the sequence they arrive
- ▶ If (for example) a serial transaction abends with a U3303, it will not be put on the suspend queue
  - It is requeued to the head of the regular queue
  - The transaction is USTOPPED
  - When transaction is started, original sequence is preserved

## ▲ With shared queues

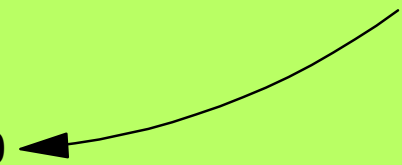
- ▶ Serialization of transaction scheduling is guaranteed only within the front-end IMS
- ▶ A serial transaction will always be scheduled only on the F-E IMS
  - Special QNAME includes IMSID
  - Guarantees serialization within that IMS
- ▶ The same transaction code, defined as SERIAL=YES in multiple IMSs, may not execute serially across the Parallel Sysplex

## Serial Transactions ...

### ▲ If SERIAL processing is important

- ▶ Define the TRANSACT as non-serial and the APPLCTN as serial
- ▶ Assign transaction to a CLASS that will only execute on one IMS
- ▶ Start one MPP region on one IMS with the transaction class specified

```
APPLCTN   PSB=SERPSB,SCHDTYP=SERIAL,...  
  
TRANSACT  CODE=SERTRAN,SERIAL=NO,MSGTYPE=(,10),...  
  
//MPPJCL  EXEC   DFSMPR,CL1=010
```



- ▶ Use the Non-discardable Message Exit (DFSNDMX0) to REQUEUE and USTOP any "pseudo-serial" transaction that abends with a U3303
  - R15 = 12 (requeue to original transaction and USTOP transaction)

## Undefined Resources

---

▲ **FINDEST (Find Destination) routines are invoked to find the destination of a message before putting it on the shared queue**

- ▶ Destination may be found
  - Transaction (local or remote)
  - Logical Terminal (local or remote)
  - MSNAME
  
- ▶ Destination may be not-found
  - Undefined in local IMS
  
- ▶ Undefined (not-found) destinations may still be valid
  - Must determine if destination is valid
  
- ▶ Undefined destinations may be dynamically created
  - DFSINSX0 (Output Creation Exit)

# Undefined Resources ...

## ▲ If destination is undefined in local IMS

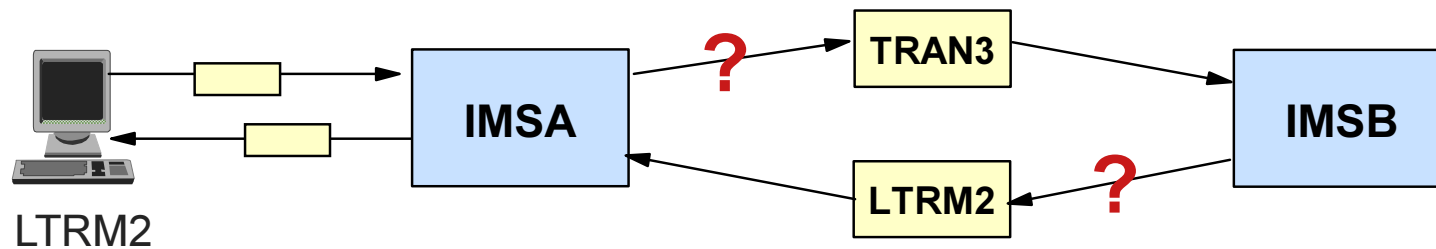
- ▶ May have been defined in other IMSs in the shared queues group
  - IMS does not know what resources are defined in other IMSs

IMSA

TRANSACTION	TRAN1
TRANSACTION	TRAN2
NAME	LTRM1
NAME	LTRM2

IMSB

TRANSACTION	TRAN1
TRANSACTION	TRAN3
NAME	LTRM1
NAME	LTRM3



## Undefined Resources ...

---

### ▲ IMSA must decide what to do with TRAN3

- ▶ Reject it
  - DFS064I Destination cannot be found or created
- ▶ Put it on queue for IMSB
  - Must determine whether TRAN3 is valid for IMSB

### ▲ IMSA can create dynamic transaction

- ▶ Output Creation Exit (DFSINSX0) can create transactions **for the purpose of putting them on the shared queue**
  - ETO not required for dynamic transactions
- ▶ If DFSINSX0 defines input message as a Transaction
  - Input is queued to Transaction Ready Queue
  - IMSB will schedule transaction and ISRT response to IO-PCB
- ▶ **Dynamic transactions cannot be executed locally (i.e. in IMSA)**

## Undefined Resources ...

---

### ▲ If IMSB tries to retrieve transaction from shared queue and input LTERM is not defined to IMSB ...

- ▶ If ETO is enabled
  - Dynamic terminal is created
- ▶ If ETO is not enabled
  - Temporary control block is created for LTERM
  - Enables program to respond to input LTERM

### ▲ If application issues call to ALT-PCB and destination not defined

- ▶ DFSINSX0 is invoked
  - May create dynamic transaction or dynamic LTERM (if ETO enabled)
- ▶ If DFSINSX0 rejects destination, call fails
  - CHNG calls gets A1 status code
  - ISRT call gets QH status code

# Security Considerations

---

## ▲ Front-end security

- ▶ Nearly all security checking is performed on the front-end
  - Signon security
  - Command authorization
  - Transaction authorization

## ▲ Back-end security

- ▶ Security environment (RACF ACEE) will be dynamically established in the back-end dependent region if needed for
  - CHNG, AUTH, and ICMD calls
  - Deferred conversational program switches
  - Environment deleted at sync point
- ▶ SMU security invoked for CMD calls issued from back-end IMS

## ▲ Transaction must be statically defined in the environment in which security is being checked

- ▶ No security checking is performed for dynamic transactions
  - Undefined resource block created by DFSINSX0



# Queue Buffer Usage

## ▲ Uncommitted messages are kept in the msg Queue Buffers

- ▶ U758 is still received if running out of DRRN
- ▶ Max available DRRN: same consideration for short and long message queue buffers:

**max DRRN=blkfct\*9999** (9999 max Qbuf number in IMS V6)

**blkfct=qbufsz/shmsgsz** (blocking factor for short msg)

**blkfct=qbufsz/lgmsgsz** (blocking factor for longshort msg)

**max qbufsz=30632**

## ▲ **Max Qbuf 9999 limit removed in IMS V7**

- ▶ 20% of QBUF expanded when QMNGR runs out of Qbuffers
- ▶ If needed will dynamic expand until QBUFMAX is reached
- ▶ If QBUFMAX not specified can expand until storage (Private/Above) is exhausted

## ▲ **Review Short and Long Queue Buff LRECL for uncommitted msg's distribution in the Queue Buffer**

- ▶ If  $\text{msg} < \text{shmsgsz}$  or  $\text{shmsgsz} < \text{msg} < \text{lgmsgsz}$  shrmmsg Qbuf used
- ▶ If  $\text{msg} > \text{lgmsgsz}$  lgmsg Qbuf and CF Staging Queue used (2nd and subsequent full Qbuf moved to the CF Staging Queue)

## Queue Space Notification Exit (DFSQSPC0)

---

### ▲ Traditional queuing

- ▶ Called when a QBuffer is allocated to a message queue data set (DRRN)
- ▶ Knows number of messages currently in use
  - Can reject insert

### ▲ In shared queues environment

- ▶ Called when a QBuffer is allocated to a data object
- ▶ Exit cannot tell how full structures are
- ▶ Parameter list passed to exit includes two new bit settings
  - **Structure is in overflow mode**
  - **Destination is in overflow mode**
    - This particular queue name is in overflow
  - Exit may use these bit settings to reject the placement of messages on the queue
    - Application gets A7 status code

# Long / Short message size and LE/DE utilization

---

## ▲ LGMSGSZ and SHMSGSZ affect LE and DE utilization

- ▶ If  $msg < shmsgsz$  then one LE and needed DE: as many DE=512 as required to save the msg.

If  $msg=600$  and  $shmsgsz=800$  then will need 1 LE and 2 DE's

- ▶ If  $shmsgsz < msg < lgmsgsz$  then one LE and needed DE: as many DE=512 as required to save the msg.

If  $shmsgsz=800$   $msg=1200$   $lgmsgsz=1024$  then will need 1 LE and 3 DE's

- ▶ If  $msg > lgmsgsz$  then one LE required for each  $lgmsgsz$  and DE as needed:

If  $msg=5120$  and  $lgmsgsz=1024$  then 5 LE and 10 DE ( $10 \times 512 = 5120$ )

## APPC/OTMA Transactions

---

### ▲ IMS V6

- IMS/ ESA V6 required **all** APPC/OTMA input messages to process on the *Shared Queues front-end* IMS system
- Consider your APPC/OTMA workload

### ▲ IMS V7 **asynchronous APPC/OTMA**

- allocate, send, deallocate - for APPC -
- commit-then-send (commit mode 0) for OTMA
- input messages and all spawned transactions can be processed by any IMS in the IMSplex

# IMS Cold Start

---

## ▲ IMS Shut Down cleanly

- No effect on the shared queue

## ▲ IMS Shut Down abnormal

- if IMS COLD START then Indoubt Messages moved from the Lock Queue to the Cold Queue (IMS MRQ to requeue them)
  - Dump comm=(dump information)  
nnSTRLIST=(STRNAME=msgstrname,(LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER))

## ▲ Unresolved UOWE (str full) lost after IMS Cold Start

- DFS1994 (introduced by PQ27818 and PQ32590) shows, at any system checkpoint if unresolved UOWE's exist and their age

## ▲ Structures, SRDS ds deleted for queue clean-up

- Non-SQ clean up (IMS Cold Start) obtained forcing Structure delete

# Reading CQS Log Records

## ▲ To print CQS log records from the logstream

```
//CQSERA10 JOB .....  
//STEP1 EXEC PGM=DFSERA10  
//STEPLIB DD DSN=IMS.RESLIB,DISP=SHR  
//SYSPRINT DD SYSOUT=A  
//TRPUNCH DD SYSOUT=A,DCB=BLKSIZE=80  
//SYSUT1 DD DSN=SYSLOG.MSGQ01.LOG,  
SUBSYS=(LOGR,IXGSEXIT),  
DCB=BLKSIZE=32760  
  
//SYSIN DD *  
CONTROL CNTL H=EOF  
OPTION PRINT EXITR=CQSERA30  
END  
/*
```

Use the LOGNAME  
specified in  
CQSSGxxx.



## MVS LOG Offload Data Set

▲ **Offload data set dynamically allocated by system logger when**

- The high offload threshold for the CF for a log stream is reached
- Recovery for a log stream is complete, and the system logger flushes all log data to DASD
- Structure rebuild occurs for the CF log stream
- The last connector to the log stream disconnects

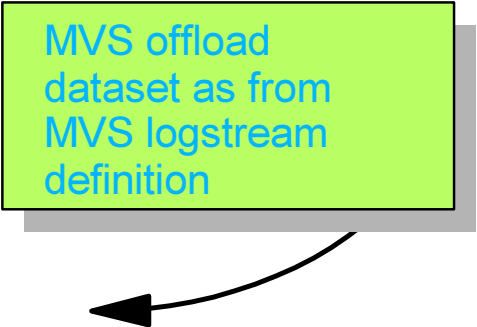
▲ **Log offload data sets automatically deleted by the logger when they no longer hold valid data**

- 2 SRDS Checkpoint

▲ **To dump MVS logstream offload dataset**

```
//job1 JOB .....DFDSS JOB ....  
//STP1 EXEC PGM=ADRDSSU,REGION=4096K,TIME=1400  
//SYSPRINT DD SYSOUT=*  
//IN DD UNIT=xxxx,VOL=SER=yyyyyy,DISP=SHR  
PRINT INDD(IN) DS(CQS.mvs.logstream.offload.A0000000.DATA) TOL(ENQF)
```

MVS offload  
dataset as from  
MVS logstream  
definition



## Review of Special Considerations

---

### ▲ **MSC has enhancements in a V6 shared queues environment**

- ▶ Processing MSC messages on any IMS in SQ Group
- ▶ Cloning IMS systems

### ▲ **SERIAL transactions must be processed on front-end**

- ▶ Serial processing not supported across Sysplex
- ▶ Can simulate serial processing with the Non-discardable Message Exit

### ▲ **Undefined resources**

- ▶ Dynamic LTERMs still require ETO
- ▶ Dynamic Transactions can be defined by Output Creation Exit
  - For purposes of putting message on queue
  - Be very careful



## Review of Special Considerations...

---

### ▲ Security is invoked in front-end

- ▶ OK, except for dynamic transactions
- ▶ Security environment built in back-end dependent region
- ▶ Build Security exit available to bypass building security environment if not needed

### ▲ Logging is different with shared queues

- ▶ IMS logs activity on each system
- ▶ CQS logs structure activity
  - Uses MVS system logger
- ▶ May require multiple logs for each transaction
  - Can be related by UOW ID

### ▲ Different IMS Cold Start approach

## Exits may have special Considerations...

---

### ▲ Some exits may have special considerations

- ▶ AOI exits (DFSAOUE0 and DFSAOE00)
- ▶ Fast path Input Edit/Routing Exit (DFSHAGU0)
- ▶ Output Creation Exit (DFSINSX0)
- ▶ Queue Space Notification Exit (DFSQSPC0)
- ▶ Front-end Switch Exit (DFSFEBJ0)
- ▶ Conversational Abnormal Termination Exit (DFSCONE0)
- ▶ Security Exits (DFSCTRN0, DFSCCTSE0, and DFSCCMD0)
- ▶ Signon/Signoff/Logoff Exits (DFSSGNX0, DFSSGFX0, DFSLGFX0)