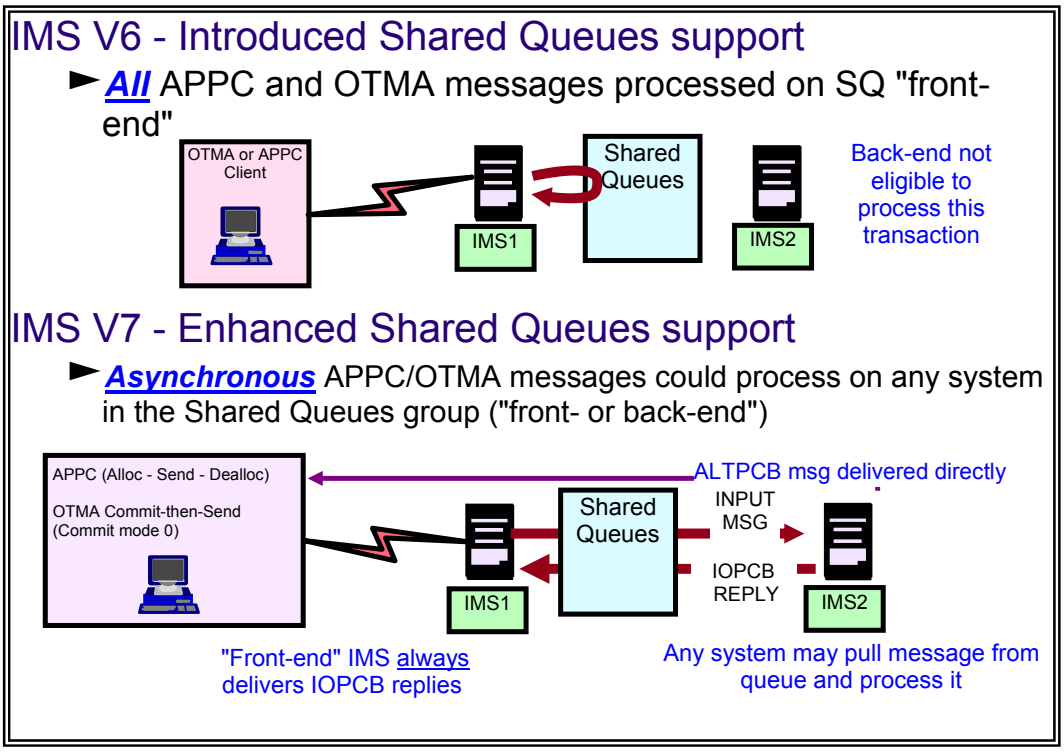
# Using the APPC/OTMA Synchronous SMQ Support with IMS V8

**Alison Coughtrie**
**alison_coughtrie@uk.ibm.com**


**Last Updated on November 11, 2004**

# APPC/OTMA Synchronous SMQ Support Overview

## Before IMS Version 8




When Shared Queues support was introduced in IMS V6, the processing of all APPC and OTMA messages (synchronous and asynchronous) was restricted to the front-end IMS. As shown in the picture, a "front-end" is the IMS to which the APPC or OTMA client is connected and a "back-end" is any other IMS in the Shared Queues group.

With the introduction of IMS V7, the support was enhanced to allow asynchronous messages through the IOPCB to be processed by any IMS in the group. This continues to be the way asynchronous messages are processed in IMS V8. For asynchronous support, IMS queues input messages to the appropriate transaction ready queue for processing by any available IMS system. IMS also stores information about the requester (tpipe token or remote LU token) along with the shared message queue (SMQ) name in the input message prefix. This information (token plus SMQ name) is used for output messages and becomes the global shared queue name for IOPCB replies.

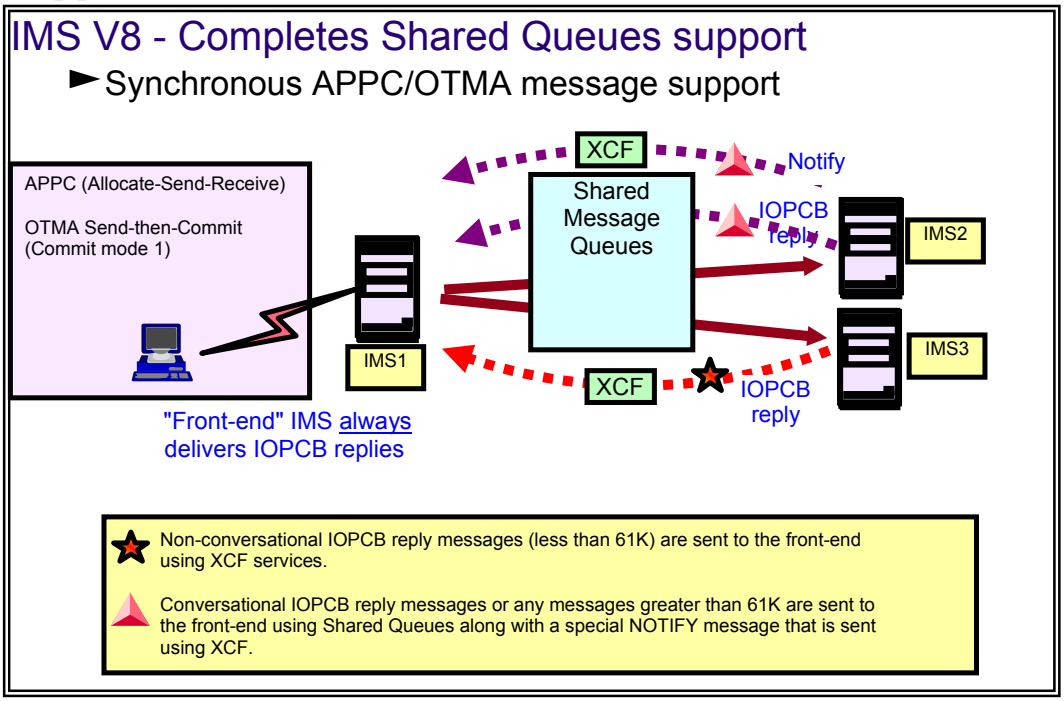
If the IMS system that picks up the message for processing happens to be the same system that received the message, then the IOPCB reply is queued to the global shared queue and the appropriate task (OTMA or APPC) is notified that it has output to process.

On the other hand, if the message is processed on a back-end system, the front-end is still required to process IOPCB replies. The IOPCB output is queued to the global shared queue but since the front-end system does not register interest in the global shared queue for potential IOPCB replies, a special process is used to notify the front-end system of these messages. This is done by creating a "notify" message in the back-end that is queued to a specialized OTMA/APPC task in the front-end, each with a

unique queue name. This message is an additional message that is associated with the IOPCB reply. For APPC, the queue name for the "notify" message is "05+DFSAPPCQ+front-end SMQ name". For OTMA, the name is "05+DFSOTMAQ+front-end SMQ name". The front-end registers interest in both these queues. The specialized OTMA/APPC task in the front-end reads the "notify" message which contains the message output queue name, and notifies the appropriate task to process the output.

ALTPCB messages are sent directly from the IMS system that processes the transaction. This system must therefore be able to establish communications with the APPC/OTMA client, meaning that all systems must be APPC and/or OTMA enabled. If a back-end IMS does not have APPC or OTMA enabled, any asynchronous APPC or OTMA output that is inserted to an alternate PCB is simply queued and not delivered until the operator issues a `/STA APPC` or `/STA OTMA` command.

## IMS Version 8 with the APPC/OTMA Synchronous SMQ Support




With IMS V8, the restriction on synchronous messages has been removed. The new capability allows any IMS in a Shared Queues group to process synchronous APPC and OTMA inbound messages, enabling the APPC/OTMA workload to be distributed amongst the Shared Queue group. In prior releases, synchronous messages had to be processed on the IMS system that maintained the connection to the client ("front-end").

APPC synchronous messages are those sent in by a verb set of Allocate-Send-Receive. OTMA synchronous messages are those that carry a commit mode of Send-then-Commit (Commit Mode 1). In either case, synchronous processing is characterized by a partner client program sending in a message and waiting for a reply. Input and output messages are non-recoverable. The IMS that receives the connection request maintains information about the connection and subsequently uses the connection to send the output reply.

When an input message is received from an OTMA/APPC partner, the receiving IMS determines whether the environment is Shared Queues capable and whether the request is asynchronous or synchronous. If it is asynchronous, then the actions described on the previous page are taken. If the message is synchronous then IMS also makes a check to see if the synchronous support is enabled. IMS stores information about the requester (tpipe token or remote LU token) along with the SMQ (Shared Message Queue) name, e.g., IMSID, in the input message prefix.

If the IMS system that picks up the message for processing happens to be the same system that received the message, then the IOPCB reply is not put on the Shared Queues but rather sent directly to the partner client prior to syncpoint processing, with the front-end holding the IMS resources until an indication of a commit or an abort. This is "business-as-usual" processing.

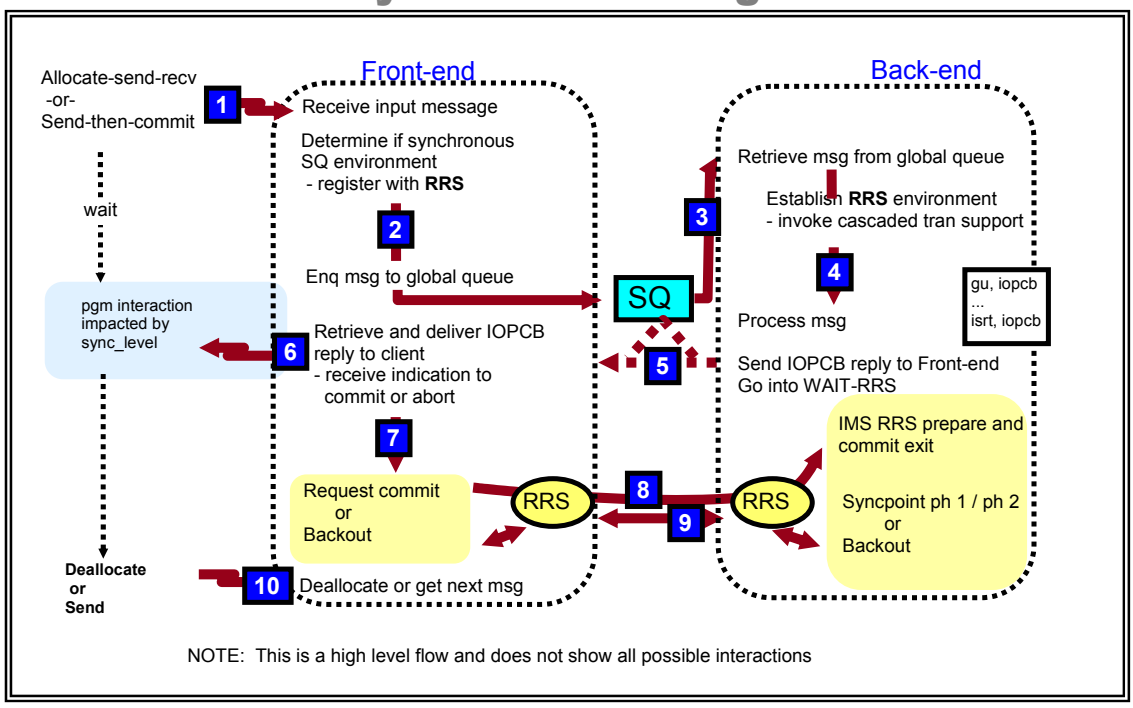
If the message is processed on a back-end system then the IOPCB reply must be routed back to the front-end IMS for delivery since it is the front-end that maintains the connection to the client. The routing is done prior to syncpoint processing with the back-end holding the IMS resources until an indication of a commit or an abort.

If a synchronous APPC or OTMA transaction running in a back-end IMS results in asynchronous APPC or OTMA output that is inserted to an alternate PCB, APPC or OTMA must be enabled on the back-end IMS. Also, the asynchronous output will be delivered to the APPC or OTMA client directly from the back-end IMS. If a back-end IMS does not have APPC or OTMA enabled, any asynchronous APPC or OTMA output that is inserted to an alternate PCB is simply queued and not delivered until the operator issues a `/STA APPC` or `/STA OTMA` command.

**Note the following:**

All conversational IOPCB reply messages and messages where all segments added together are greater than 61K, are routed through the Shared Queues with a special "notify" sent via XCF. A specialized OTMA/APPC task in the front-end reads the "notify" message which contains the message output queue name, and notifies the appropriate task to retrieve the message from the Shared Queues. All non-conversational IOPCB reply messages less than 61K are sent to the front-end using XCF services.

## General Flow of a Synchronous Message



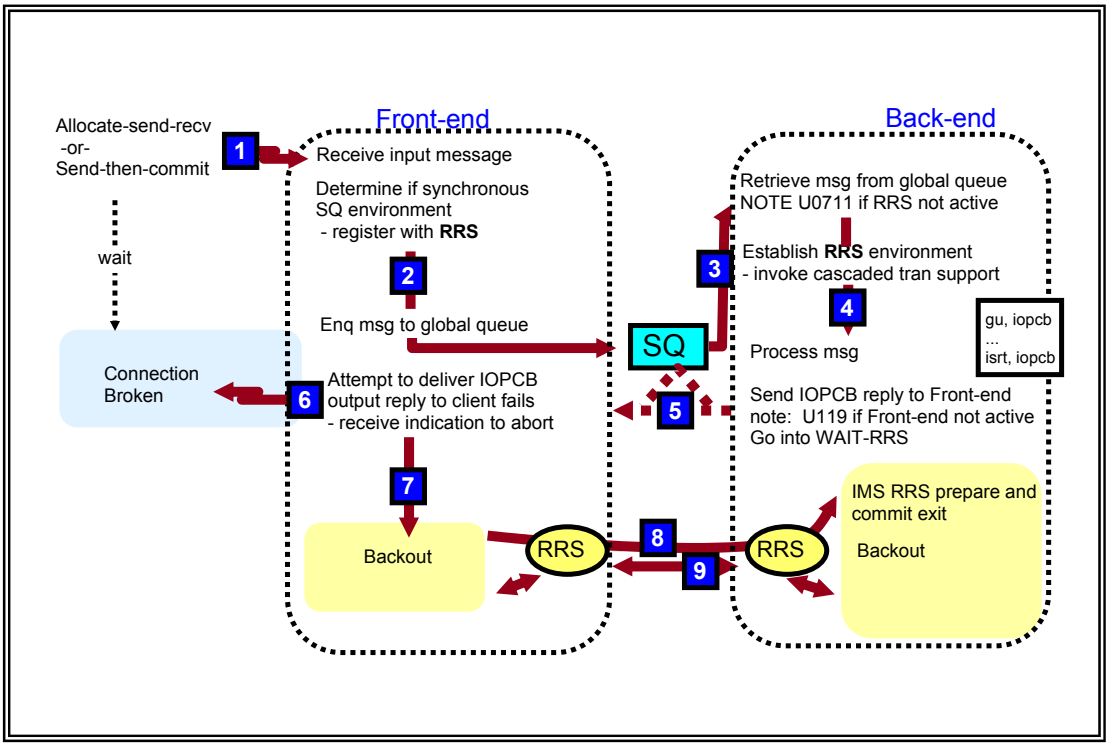

NOTE: This is a high level flow and does not show all possible interactions

This visual provides a general flow of a synchronous message as it is processed in an IMS Shared Queues environment:

1. When an input message is received, IMS determines if the message is synchronous, checks to see if the Shared Queues capability is enabled, and invokes RRS callable services to establish the RRS working environment.

2. The input message is placed on the global Shared Queues to be processed by an available IMS, i.e., an IMS where the transaction is defined and has an available message region. In this example, the Back-end system is immediately available to process the transaction.
3. The back-end IMS invokes RRS callable services to establish the RRS environment and perform message synchronization.
4. The message is processed in the back-end and the IOPCB reply sent to the front-end. All IMS resources are held pending syncpoint processing until either a commit or backout indicator is received.
5. Depending on the size of the IOPCB reply and whether or not this is a conversational transaction, the message is either sent via XCF services or routed through Shared Queues.
6. The front-end IMS sends the message and interfaces with the partner client following the rules of sync_level processing - either None, Confirm or Syncpoint.
7. Based on the success/failure of partner interaction, the front-end IMS invokes RRS to either commit or backout.
8. The RRS commit/backout is communicated to the back-end for the corresponding commit/backout.
9. RRS on both sides provides the support for the synchronization of commit/backout.
10. At the completion of commit/backout, the front-end IMS interacts with the partner program to either terminate the connection or get the next message.

## General Flow of a Synchronous Message Reply which cannot be Delivered




This visual provides a general flow of a synchronous message as it is processed in an IMS Shared Queues environment and the response cannot be returned by the front-end:

1. When an input message is received, IMS determines if the message is synchronous, checks to see if the Shared Queues capability is enabled, and invokes RRS callable services to establish the RRS working environment.
2. The input message is placed on the global Shared Queues to be processed by an available IMS, i.e., an IMS where the transaction is defined and has an available message region.

3. In this example, the Back-end system is immediately available to process the transaction. The back-end IMS invokes RRS callable services to establish the RRS environment and perform message synchronization.
4. The message is processed in the back-end and the IOPCB reply sent to the front-end. All IMS resources are held pending syncpoint processing until either a commit or backout indicator is received.
5. Depending on the size of the IOPCB reply and whether or not this is a conversational transaction, the message is either sent via XCF services or routed through Shared Queues.
6. The front-end IMS cannot send the message to the partner client.
7. Based on the failure of partner interaction, the front-end IMS invokes RRS to backout.
8. The RRS backout is communicated to the back-end for the corresponding backout.
9. RRS on both sides provides the support for the synchronization of backout.

## "Synchronous" Messages

### The meaning of "synchronous" has not changed

- Partner waits for IOPCB reply message or
  - DFS2082 message if the transaction terminates without iopcb reply
  - New message DFS2224 if tran processes in a back-end that abends
- Data is sent by the front-end before IMS commit processing
  - When message is processed on the front-end
    - Processing is like that in a non-SQ environment
  - When message is processed on a back-end IMS
    - Both front-end and back-end are part of the commit

### Supports

- Sync_levels: None, Confirm, Syncpoint
- All transaction types except APPC CPI-C Driven

### Uses RRS (Resource Recovery Services) - z/OS V1.2 or later

- Allows synchronization of message-processing and the associated commit across IMS systems

It is important to note that the meaning of "synchronous" APPC/OTMA message processing has not changed in this environment. The rules of processing that apply in a non-Shared Queues implementation continue to apply when Shared Queues synchronous processing is enabled. By definition of "synchronous", the partner program continues to wait for a reply message from IMS. This reply can take the form of an IMS application IOPCB reply or a DFS error message. The DFS2082 message which is sent in a non-SQ environment when IMS transaction processing terminates without an IOPCB reply continues to be sent with synchronous SQ enablement. When the front-end IMS processes the input message, the processing is similar to that in non-SQ with the exception that the RRS environment is established. When processing occurs on the back-end, both IMS systems are involved in the RRS commit/backout process. If a transaction is processing in a back-end system and the back-end fails, the front-end is notified and a message is sent to the remote client which releases the synchronous wait: DFS2224I TRANSACTION ON A BACK-END SYSTEM ABENDED.

Input messages can invoke all transaction types except for APPC CPI-C driven programs. (APPC messages processed by explicit calls are not passed through the message queues.)

APPC/OTMA Shared Queues support is based on RRS services. It is RRS that provides the environment and allows synchronization of message processing across multiple IMS systems.

## Program-to-Program Switches

Synchronous messages and pgm-to-pgm switches

- (sync_level=syncpoint) distributed transactions cannot issue program-to-program switches
- (sync_level=none | confirm) transactions can issue program-to-program switches but all involved transactions must run in the same IMS
  - Except for conversational transactions
    - Any transaction in the conversation can run on any IMS in the SQ group
  - Unique situation where a transaction running on a back-end IMS spawns multiple transactions (at least one local and one remote)
    - Local transactions must run in the front-end IMS
    - Remote transactions are sent across the MSC link to the remote IMS. Any responses sent back to the SQ group are sent to the front-end IMS.

It is important to determine whether or not the transactions participate in program-to-program switches.

Program-to-program switching is not supported for messages that are sent in with a sync_level of syncpoint. This restriction is true even in a non-Shared Queues environment.

For messages sent in with a sync_level of NONE or CONFIRM, all transactions involved in the program-to-program switches, as a general rule, must run in the same IMS (front-end or back-end) that processes the first message. So if Tran A (synchronous) spawns B (synchronous) and C (asychronous), A can run on any system (does not have to be the input system) but B and C will run on the same system as A.

There are some exceptions to this restriction. Conversational transactions, whether immediate or deferred, can run in any IMS in the Shared Queues group. The other exception involves a unique situation where a message that processes in a back-end IMS issues multiple program-to-program switches - one or more to a local transaction(s) that will be processed in the Shared Queues environment **and** at least one to a remote transaction that will be sent across an MSC link. In this situation, the local transaction(s) must process in the front-end IMS, i.e., the IMS which has the connection to the APPC or OTMA partner. It should also be noted that any replies that come back across the MSC link from the remote IMS system will be sent to the front-end IMS for delivery.

## Distributed Two Phase Commit Transactions

The following restrictions apply to distributed syncpoint transactions using multiple transaction two phase commit transactions (meaning that the client sends more than one transaction to IMS (using SYNCLEVEL=SYNCPOINT) and all the transactions are within one commit scope).

1. If the database locks need to be passed to the next transaction then the transaction has to run on the same IMS system. The database locks cannot be passed between IMS systems within an IMSPLEX. The user needs to ensure this, IMS cannot know if the database locks need to be passed.
2. If the transactions need to run on the same IMS system (for example because the database locks needs to be passed) then shutdown could hang if the first transaction was processed and the second transaction is queued to the message queue. Shutdown will prevent the second transaction from being processed but the first transaction will wait in

commit status for the second transaction. Time-out values at the client will abend the first transaction after the specified time-out value and shutdown will complete. The /DIS A REG. command will display the dependent region in WAIT-RRS status. /STO REG ABDUMP can be used to terminate the dependent region.

3. The current limit for front-end RRS TCBs is 40. This could lead to "dead-locks" in the case of the excessive use of multi-tran two phase commit transactions for APPC. One RRS TCB is used for every transaction within the commit scope for APPC. This means if the client sends 10 transactions to one front-end IMS within one commit scope then 10 RRS TCBs are needed at the front-end IMS. (The back-end uses the dependent region TCBs, so doesn't have this same restriction. The RRS TCBs are used at the front-end because when IMS issues RRS commit, RRS will wait the TCB until the commit is completed. This could take a very long time because the back-end needs to complete Phase 1. If this were done using the RLM TCB then the performance would suffer. This is only an APPC consideration as for OTMA the client has to issue the commit for protected conversations. For APPC it is IMS that issues RRS commit.)
4. IMS cannot detect transactions within the same commit scope if they are using different XIDs. Therefore the database locks will not be passed in this case. (This restriction applies to all protected conversations not just those using Shared Queues.)

## Error Conditions

Error Conditions

- If partner client is unavailable and output reply cannot be sent
  - Abend U0119 - when transaction is processed on the front-end
    - Transaction is not stopped
    - DFSCMUX0 - user exit can re-route output
  - Take Backout - when transaction is processed on the back-end
    - Transaction is backed out but does not get abend
    - DFSCMUX0 - user exit can re-route output
- Additional consideration for the back-end
  - Abend U0119 - Any XCF errors that occur when sending output to the front-end
    - Transaction is not stopped
- Abend U0711 - Any RRS errors that occur before sending output
  - Transaction is stopped

When an output IOPCB reply is sent, it is always sent before syncpoint. If the transaction that replies to the IOPCB executes on the front-end and the reply cannot be sent, the default action is to abend the transaction with a U0119 and discard the output reply. User exit DFSCMUX0 (Message Control/Error Exit Routine) can be implemented to change the default action. The exit can change the default abort action (except for sync-level of syncpoint which must continue with the abort). This is the way errors with synchronous message replies have been processed for several releases of IMS. If the transaction that replies to the IOPCB executes in a back-end IMS and the front-end cannot deliver the message, an RRS Take Backout is issued which forces backout processing to occur on the back-end. The back-end aborts the transaction with U0711-1E. It is of note that although backout occurs, the transaction does not experience an abend condition. As in the front-end, DFSCMUX0 can be used to change the default action. If the transaction is executed on a back-end system and the client deallocated the conversation

then the DFSCMUX0 exit gets driven and is able to re-route the message. If re-route is successful then the front-end will issue RRS-commit.

As mentioned earlier, non-conversational transaction reply messages that are less than 61k in length are sent to the front-end IMS using XCF services. If the back-end IMS receives an XCF error when attempting to send the reply message to the front-end then a U0119 abend occurs, the output message is discarded and the remote client notified. DFSCMUX0 is not driven on the back-end system if the XCF send to the front-end fails (for the reasons explained below). There are two reasons why the XCF send to the front-end may fail: 1) The front-end is gone (in-active). An inactive front-end is treated as an IMS internal error. (In a non-sysplex environment when IMS abends then DFSCMUX0 is not be called). In this case, if DFSCMUX0 were to reroute the response message to another destination then syncpoint would still fail because the UR in RRS is in backout needed status. The parent UR is no longer there and the child is not allowed to make decisions. That's how RRS works. 2) XCF send fails and the front-end is still up. In this case the front-end (parent) has to issue the commit/abort. If the DFSCMUX0 exit were to reroute the response message then syncpoint would wait until the front-end issues commit/abort. Because the back-end couldn't inform the front-end (XCF fails) there will be no commit/backout until the front-end or back-end shuts down. The dependent region/s as well as the APPC conversation would hang. Keep in mind that the XCF send failure at the back-end only occurs for transactions that are already scheduled in the MPP. This shouldn't be very often. The majority of the transactions will probably abend with a U0711 because GU cannot cascade the UR. /STO APPC could be issued a few minutes before shutting down IMS. This will allow conversations that are already active to continue, but new conversations would be rejected.

Anytime an RRS error is received before sending output, a U0711 is issued and the transaction is stopped.

## "Monitoring" Synchronous Support

### Synchronous support can be disabled during message processing

- For example, RRS stopped on one system
  - Synchronous APPC/OTMA transactions processing in back-end IMS systems are abended with U0711

### DFS Messages

- System console messages
  **DFS0653I/DFS0698W/DFS0548A** when RRS is activated/deactivated
- Two new messages for APPC/OTMA SMQ enablement status:
  **DFS2089I** and **DFS2088I**

### Operator command - /DISPLAY ACTIVE

- Shows status of synchronous Shared Queues enablement

```
DFS000I APPC/OTMA SHARED QUEUE STATUS - LOCAL=INACTIVE
  GLOBAL=INACTIVE
```

It is possible for the synchronous support to be disabled while a message is being processed. The fact that APPC/OTMA SMQ Enablement gets disabled doesn't mean that IMS cannot send the message to the front-end if the front-end is still there. IMS will be able to deliver the message to the client and commit the data even if APPC/OTMA SMQ Enablement got disabled because another IMS was not able to participate. If APPC/OTMA SMQ Enablement gets disabled, new incoming transactions will be queued with affinity. IMS will always try to commit the data and will not check the status of APPC/OTMA SMQ Enablement after the enqueue.

Existing messages that are sent to the system console when RRS is activated or deactivated, continue to be valid in IMS V8. These have a slightly different format now due to the RRS=Y/N option on the IMS control region. The DFS0548A message is new. The messages include:

- DFS0698W PROTECTED CONVERSATION PROCESSING NOT ENABLED – RRS=Y NOT SPECIFIED / OPERATOR DECISION
- DFS0653I PROTECTED CONVERSATION PROCESSING WITH RRS ENABLED
- DFS0548A RRS NOT ACTIVE BUT RRS=Y SPECIFIED - REPLY: RETRY, CONTINUE OR CANCEL

There are two new messages to assist operators:

- **DFS2089I APPC/OTMA SMQ Enablement Active**

**Explanation:** The APPC/OTMA Shared Message Queue Enablement is active.

**System Action:** IMS will allow every member within the IMSPLEX to process APPC/OTMA messages.

- **DFS2088I APPC/OTMA SMQ Enablement Inactive. Reason = xxx.**

**Explanation:** The APPC/OTMA Shared Message Queue Enablement is inactive.

xxx is the reason code. The following reason codes are possible:

| Code | Meaning |
|---|---|
| 004 | This is a non-shared queues environment |
| 008 | The keyword MINVERS is set below minimum support level. Ensure that all IMS systems within the IMSplex are version 8 or higher and set MINVERS(8.1) or higher. |
| 012 | IMS has been started on a z/OS level lower than 1.2. |
| 016 | RRS is not active. |
| 020 | A member joined the group and cannot support APPC/OTMA SMQ Enablement. |
| 024 | Another member dropped support for APPC/OTMA SMQ Enablement. |
| 028 | IMS disconnected from RRS. |
| 032 | AOS=N was specified in the DFSDC*xxx* member. |
| 036 | RRS=N has been specified as a start-up parameter. |

**System Action:** IMS will force every APPC/OTMA message to the front-end system.

**Operator command: /DISPLAY ACTIVE**

The /DISPLAY ACTIVE command has been enhanced to display the local and global status of the synchronous SQ enablement. The local status tells if a particular IMS is able to support the synchronous APPC/OTMA capability. The global status tells if the enablement is active within the shared message queue group.

## Performance

RRS Performance

- ► Directly impacts synchronous APPC/OTMA processing in IMS
  - ● Review logstream performance (Coupling Facility versus DASD only)
- ► Review “XCF Tuning for IMS Running in a Parallel Sysplex Environment”
  - ● IMS Newsletter, Spring 2004
    http://www-306.ibm.com/software/data/ims/quarterly/index.html

When the synchronous support is enabled, it is worth noting that RRS performance directly impacts IMS performance. The primary areas to consider are the dispatching priority of the address space as well as the setup of the logstream. Refer to: z/OS MVS Setting Up a Sysplex (SA22-7625) for more information.

z/OS MVS Programming: Resource Recovery (SA22-7616) has detailed information on RRS.

## RRS "Affinity"

"Affinity"

- If IMS fails and RRS does not, if there are outstanding units of recovery, you cannot restart IMS on a different image
  - If there are NO outstanding units of recovery, you may restart IMS anywhere in the sysplex.
- If IMS fails and RRS also fails, then you may restart IMS on another image even if there are outstanding units of recovery.
- To move IMS back, you must shutdown IMS cleanly such that all units of recovery are resolved properly or RRS won't allow it to be moved

With the Synchronous SMQ support RRS is used as the syncpoint coordinator for the transaction processed at the backend system, and coordinates the two-phase commit process. From the time that the resource manager issues its response to the PREPARE request (the completion of Phase 1), to the time it receives a COMMIT or ABORT request from the RRS, units of recovery are said to be *in-doubt.*

If IMS fails and RRS does not, if there are outstanding (in-doubt) units of recovery, you cannot restart IMS on a different image. If there are NO outstanding units of recovery, you may restart IMS anywhere in the sysplex.

If IMS fails and RRS also fails, then you may restart IMS on another image even if there are outstanding units of recovery. So if the image fails, all programs are dead, so you can move IMS anywhere in the sysplex (unless you restart the image and RRS before you restart IMS).

To move IMS back, you must shutdown IMS cleanly such that all units of recovery are resolved properly or RRS won't allow it to be moved.

# Enabling Synchronous APPC/OTMA SMQ Support

## Prerequisite Software

- Synchronous APPC/OTMA SMQ Enablement - APAR PQ66303
- All IMS systems in the Shared Queues group must be V8 or later
  - RECONs must specify MINVERS(8.1) or higher
- z/OS V1 R2 or later
  - Plus RRS APAR OW50627 (check PSP bucket for additional prerequisites)
  - ***Be as current as possible with maintenance for RRS***
- AOS=Y/N/F
  - Must be specified in DFSDCxxx member. Default is N, INACTIVE
- All z/OS environments with an IMS in the Shared Queues group must have RRS enabled
  - RRS=Y must be specified as a Control Region execution parameter
  - If RRS is not active or is deactivated in one system, synchronous support for the group is disabled (unless AOS=F)

The APPC/OTMA synchronous Shared Queues support is enabled when the DFSDCxxx parameter specifies, AOS=Y, and a combination of prerequisite conditions is satisfied. These include: a Shared Queues environment with a minimum level of IMS V8 for all the IMS systems in the group, a DBRC RECON specification of "MINVERS(8.1)", associated operating system levels at a minimum of z/OS V1.2 along with RRS APAR OW50627 (check PSP bucket for most recent service) and active RRS (Resource Recovery Services) support on each of the appropriate z/OS systems. Additionally RRS=Y must be specified on the IMS Control region.

Every time an IMS joins the SMQ, all members communicate their current synchronous SQ status (based on adherence to the minimum requirements) to enable/disable the function as necessary. Also, if one of the IMS systems connects or disconnects from RRS, i.e., RRS is enabled/disabled, all members of the group are informed. If all IMS systems and environments meet the requirements, the synchronous support is enabled. If one of the members is unable to support the enablement, it is disabled for the group.

All members of the shared queues group should specify the same value for RRS=. The function will be enabled (or not) based on the settings of the first IMS to connect.

DFSDCxxx PROCLIB member, parameter: AOS=Y/N/F

> Specifies whether the synchronous APPC/OTMA SMQ Enablement function should be active (Y) or inactive (N). The default is inactive (N). APPC/OTMA SMQ Enablement allows IMSPLEX users to execute transactions originated by APPC or OTMA on a back-end system. If force (F) has been specified then the function will be active even if another member is not able to activate synchronous APPC/OTMA SMQ Enablement.
>
> **NOTE:** Specifying N will deactivate APPC/OTMA SMQ Enablement for all the members within the IMSPLEX except for members specified F (force). APPC/OTMA SMQ Enablement uses RRS Multi-System Cascaded Transaction (MSCT) support to synchronize between the IMS systems. If RRS is not active on one system then the systems which have specified F (force) will still queue incoming transactions without any affinity. Therefore if the system without RRS tries to process one of these transactions it will abend the application with U0711. This option was added to address a specific customer scenario where the front-end has no databases, and if one back-end RRS goes down then all the distributed sync point transactions would have affinity to the front-end which is not able to process them.

## Diagnostics

RRS Component Trace
- ►START by issuing 'TRACE CT,100M,ON,COMP=SYSRRS'
  - •Rxx,OPTIONS=(EVENT(ALL))
- ►STOP by issuing 'TRACE CT,OFF,COMP=SYSRRS'

RRS Panels
- ►Look at the status of URs and RMs in ISPF

IMS Log records
- ►Type 4098 for RRS/MVS logname
- ►Type 67D0 for U711 abends
- ►5615, 5616

IMS /DIS UOR command
- ►Shows outstanding UORs that IMS knows about

/TRA SET ON TABLE RRST
- ►Show the major IMS/RRS invocation and RC
- ►Trace abnormal RRS events in IMS

Reference
- ►MVS Programming: Resource Recovery (SA22-7616)
- ►IMS Failure Analysis Structure Tables (FAST) publication for ABENDU0711 return codes

Usage of the RRS panels is described in "Managing RRS" in MVS Programming: Resource Recovery.

The /DISPLAY UOR command, when issued at the back-end IMS, has been updated to display information for non-protected transactions running at the back-end IMS in the shared queues group.

A new RRS trace table has been added to record the flow of control and the activities through the RRS components. To activate or deactivate this new trace table, use the new RRST parameter on the /TRACE command (for example, /TRACE SET ON TABLE RRST). When you specify OPTION LOG, IMS writes the trace externally as type X'67FA' records. Additionally RRST Trace Table formatting will be enhanced (via APAR PQ89597) with the addition of a new formatting routine DFSERA61 which is transparent to the end user. This new routine is called by Log Print Utility (DFSERA10) exit DFSERA60 or directly using EXITR=DFSERA61 parameter when an RRST/APPC/OTMA trace record is detected.

If a /DIS A REG command shows the dependent region in WAIT-RRS/PC status, a /STO REG ABDUMP command can be issued to terminate the region.

# Maintenance

## Recommended IMS Maintenance

The actual APPC/OTMA SMQ enablement APAR is PQ66303 and it's PTF was made available on 28/05/04. The maintenance below itemizes the APARs that were identified during the development and testing of the facility and have been included here for completeness. Recent CBPDO's will incorporate these, and it's recommended ***that you obtain a current PDO before implementing the support. PSP buckets should always be checked.***

| V8 APAR | V8 APAR/PTF Number | PDO | Description |
|---|---|---|---|
| PQ58092 | PQ58092/UQ63132 | 0209 | The fix description DFSCMS00 was changed to correctly add the front end affinity to the sq queue name when processing APPC and OTMA synchronous messages. |
| PQ58197 | PQ58197/UQ63222 | 0212 | Application program on BE didn't get u711 when FE crashed |
| PQ59768 | PQ59768/UQ67768 | 0228 | This APAR fixes various problems for APPC/OTMA SMQ Enablement. |
| PQ59981 | PQ59981/UQ79902 | 0339 | OTMA synchlevel setting environment |
| PQ60753 | PQ60753/UQ67291 | 0226 | Dump formatting enhancement for synchronous APPC/OTMA shared queue enablement |
| PQ61886 | PQ61886/UQ70044 | 0240 | The IMS/RRS failure notification code for APPC/OTMA V8 SPE |
| PQ62033 | PQ62033/UQ68068 | 0229 | This APAR adds the support of using multiple TCBs for the RRS commit/backout service. |
| PQ62612 | PQ62612/UQ70770 | 0243 | IMS APPC/OTMA Shared Queue enhancements for various 711 abends |
| PQ62756 | PQ62756/UQ69903 | 0239 | IMS APPC/OTMA status inconsistency enhancement |
| PQ62873 | PQ62873/UQ70789 | 0243 | RRS= parm |
| PQ63293 | PQ63293/UQ73798 | 0306 | DFS6LUS1 ABEND0C4 during /DEQ LUNAME command (resolve 757-47 problem) |
| PQ63352 | PQ63352/UQ70125 | 0240 | IMS abended with 0C4 in DBFDSYN0 while tuning OTMA trans |
| PQ63525 | PQ63525/UQ69298 | 0236 | ABENDS0C4 DFSXCF00 after installation of UQ68071 |
| PQ63835 | PQ63835/UQ75030 | 0313 | IMS dependent region, processing APPC/OTMA input hangs |
| PQ65832 | PQ65832/UQ71994 | 0250 | Resource recovery service trace (RRST) added to IMS V8 |
| PQ66303 | PQ66303/UQ88526/7 | 0405 | V8 APPC/OTMA Shared Queues Enablement APAR |
| PQ66549 | PQ66549/UQ75105 | 0313 | U711 DFSECP20 program switch from IFP to BMP/MPP<br>U108 DFSECP10 TPIPE serialization on back end /SECURE OTMA REFRESH TMEMBER |
| PQ67715 | PQ67715/UQ72828 | 0302 | Front-end IMS ABEND0E0-34 in sync APPC/OTMA SMQ environment |
| PQ68008 | PQ68008/UQ74169 | 0309 | ABEND0C4 in DFSAOSW0 when the BE system processed fastpath transaction |
| PQ68321 | PQ68321/UQ72910 | 0302 | ABENDS0D6 in DFSRRSI0 |

| V8 APAR | V8 APAR/PTF Number | PDO | Description |
|---|---|---|---|
| PQ68365 | PQ68365/UQ72449 | 0251 | ABEND0C4 IN DFSYLUS0 processing an OTMA message from the previous front end IMS execution |
| PQ68405 | PQ68405/UQ73579 | 0307 | AWE not released in DFSAOSW0 |
| PQ68439 | PQ68439/UQ72829 | 0302 | When RRS abended unexpectedly, the IMS control region abends with ABENDS0D6 |
| PQ69762 | PQ69762/UQ74038 | 0307 | IMS control region abended with a S0C4 after immediate program-to-program when processing APPC messages in APPC/OTMA SMQ enablement environment. |
| PQ69947 | PQ69947/UQ73794 | 0306 | ABEND 711 in DFSECP10 running FP APPC/OTMA trans in SMQ environment |
| PQ70560 | PQ70560/UQ75888 | 0317 | MSGDFS070 not returned on reply-Q for OTMA when CQS is down (from V7 field APAR PQ69086) |
| PQ71275 | PQ71275/UQ76487 | 0319 | U711-1E DEADLOCK, ROLB MODE=MULTI w/ APPC/OTMA enablement, /DIS A shows wrong value |
| PQ72628 | PQ72628/UQ76853 | 0321 | U711-1D in DFSTMS00 in FE IMS OTMA processing of two protected conversations in the same unit of work. |
| PQ72948 | PQ72948/UQ77918 | 0328 | U711 in dependant region after synchronous APPC transaction abended on backend.<br>FE S0C1 Control Region Abend DFSRRSI0 – trace table wrap. |
| PQ73530 | PQ73530/UQ79088 | 0332 | Multiple TCB algorithm. Many DFS1959E console messages reason code=0606 and 1224 after cancelling CQS 729 after cancel rrs on backend. OTMA WAIT-DL/I |
| PQ73679 | PQ73679/UQ76812 | 0321 | DFS1959E MSG reason code 711 when IMS starts up |
| PQ73895 | PQ73895/UQ79704 | 0337 | DFS2146 error on CTC link stopped if OTMA tran |
| PQ74559 | PQ74559/UQ844440 | 0407 | The global queue count on an OTMA /DIS TMEMBER XXXX TPIPE YYYY QCNT erroneously shows zeroes. |
| PQ75350 | PQ75350/UQ79092 | 0332 | MSC-OTMA output processed but not sent to FE. 0C4 DFSYLUS0 |
| PQ75360 | PQ75360/UQ78676 | 0330 | 757-SC037 after /EXIT, APPC protected conversation. U757-011 in dep. region causes U113 of IMS CTL region |
| PQ75517 | PQ75517/UQ80671 | 0341 | XRF – invalid XCF member name used in APPC/OTMA enablement RRST trace via DFSVSMX member DFSCMUX0 |
| PQ76238 | PQ76238/UQ78905 | 0332 | Zero timestamp in DFS555 message |
| PQ76446 | PQ76446/UQ83080 | 0402 | XRF alternate hang connecting to RRS after RRS cancelled on active |
| PQ76713 | PQ76713/UQ79817 | 0337 | ENQCT of rerouted destination via DFSCMUX0 not incremented when BE |
| PQ76833 | PQ76833/UQ82950 | 0401 | /DIS UOR limited to protected conversations |
| PQ77043 | PQ77043/UQ91064 | 0407 | Front-end IMS system abends with u0729 when back-end system sends a zero length message to the front-end. |
| PQ77299 | PQ77299/UQ79816 | 0337 | U711 instead of U126 in FP – DFSTMR00 |
| PQ78383 | PQ78383/UQ80852 | 0344 | Abend S058 IMS CTL Region after RRS cancel |
| PQ78434 | PQ78434/UQ83607 | 0402 | DFS2089I message truncated |

| V8 APAR | V8 APAR/PTF Number | PDO | Description |
|---|---|---|---|
| PQ78670 | PQ78670/UQ83196 | 0401 | U711-20 OTMA abend fast path |
| PQ78761 | PQ78761/UQ80701 | 0341 | OTMA CM1 message is processed in a back-end IMS, the ALT-PCB output is NOT put on the hold queue with IMS Connect V1 and V2 |
| PQ78851 | PQ78851/UQ82016 | 0347 | S0C4 in DFSYSTO0 |
| PQ78862 | PQ78862/UQ81160 | 0344 | FP shared EMH - large PITB table search |
| PQ78897 | PQ78897/UQ81126 | 0344 | 729-10 in DFSSPM50 |
| PQ79168 | PQ79168/UQ85769 | 0419 | PSEUDO ABENDU0830 R15=00000004 when a remote transaction from an APPC/OTMA client is processed on a back end (BE) shared queues (SQ) IMS. |
| PQ80036 | PQ80036/UQ83164 | 0401 | 78430 0C4 DFSAOSW0 |
| PQ81351 | PQ81351/UQ83563 | 0402 | Sync protected APPC conv causes MPP U711 |
| PQ82293 | PQ82293/UQ83987 | 0407 | S0C4 in DFSSPM00 during DFSPOOL REL call |
| PQ83885 | PQ83885 | | APPC/OTMA SMQ enablement continues with GLOBAL=active when a second IMS with RRS=NO joins the group |
| PQ84179 | PQ84179/UQ86129 | 0412 | ABENDU729 in DFSSPM00 after call to DBFHIEL0 |
| PQ84546 | PQ84545/UQ90195 | 0407 | IMS XRF alternate system ABENDS40D due to private storage being exhausted |
| PQ84903 | PQ84903 | | ABENDU729-18 ABEND using APPC/OTMA SHARED QUEUES enablement |
| PQ87812 | PQ87812/UQ88450 | | MPP region loops on backend IMS after processing synchronous OTMA message from frontend IMS in shared queue environment. |
| PQ89597 | PQ89597 | | RRST Trace Table formatting is enhanced with the addition of a new formatting routine DFSERA61 which is transparent to the end user. This new routine is called by Log Print Utility (DFSERA10) exit DFSERA60 or directly using EXITR=DFSERA61 parameter when an RRST/APPC/OTMA trace record is detected. |
| PQ88956 | PQ88956/UQ91028 | O407 | RRST trace table enhancements |

## Recommended z/OS Maintenance

The APPC/OTMA Synchronous SMQ enablement support requires the use of the RRS Cascaded Transaction support in z/OS (OW50627 plus subsequent service). The maintenance below itemizes the APARs that were identified during the development and testing of the facility and have been included here for completeness. Recent CBPDO's will incorporate these, ***and it's recommended that you obtain a current PDO before implementing the support. PSP buckets should always be checked.***

| APAR/PTF Number | Product | FMID | Description |
|---|---|---|---|
| **z/OS 1.2** | | | |
| OA01584/UA01035 | 5752SCCTX | HBB7705 | 0C4 IN CTXRSWCTL |
| OA01876/UA01589 | 5752SCRRS | HBB7705 | Unexpected RC from ATRCMIT |

| APAR/PTF Number | Product | FMID | Description |
|---|---|---|---|
| OA01877, OA01879/ UA01607 | 5752SCRRS | HBB7705 | Unexpected Error Dumps during Restart / SPID Logging error |
| OA01916/UA02218/UA02219/UA022220 | 5695DF106 | HDZ11E0<br>HDZ11F0<br>HDZ11G0 | media manager returns twice for same call abend0c4 or other while preparing for or after second return. ow55469 applied. |
| OA02137/UA01596 | 5752SCRRS | HBB7705 | '0' output parms from atrbeg |
| OA02703/UA03794 | 5752SCXCF | HBB7705 | irlm delay. msgdxr168i msgdxr167e were issued due to delay in delivery by xcfas. lost usage of some of the mmdc mmdcs |
| OA04573/UA05762 | 5752SCRRS | HBB7705 | rrs latch contention for sys.atrbccdp |
| OA04771/UA06421 | 5752SCRRS | HBB7705 | contention showing deadlock for the she latch and the shat latch in atrxmscb |
| OA05073/UA08736 | 5752SCCTX | HBB7705 | latch for sys.atrurcpo not released when cntxinuse remains on not allowing the end context to complete |
| OA06191/UA08705 | 5752SCRRS | HBB7705 | task waiting on cntxinuse in atrbmecy leaving ur in forgotten (fgn) status. |
| OA07084 | 5752SCACB | HBB7705 | abend0c4 in isglrels because atbvsta attempted to release a garbage latch token. |
| OW50627/UW93830 | 5752SCRRS | HBB7705 | Rollback RRS Subordinate Failure notify support |
| OW54120/UW93843 | 5752SCRRS | HBB7705 | Rollback RRS Subordinate Failure notify support |
| OW55407/UW95847 | 5752SCRRS | HBB7705 | CTrace Updates |
| OW55775/UW96159 | 5752SCRRS | HBB7705 | Base Reg Usage / Overlay Error |
| OW56323/UA02633<br>OW56325/UA02633<br>OW56326/UA02633<br>OW56327/UA02633<br>OW56329/UA02572<br>OW56330/UA02572<br>OW56492 /UA02572<br>OW56526/UA02572<br>OA02517/UA02644<br>OA02518 /UA02656<br>OA02519/UA02667 | 5752SCRRS | HBB7705 | Misc Fixes for IMS reported problems |
| OW57349/UA02058 | 5752SCRRS | HBB7705 | RRS Hang while holding SYSZATR restart resource while rebuilds are occurring |
| OW57542/UW96212 | 5752SCRRS | HBB7705 | Additional Modules for OW55775 |
| OW57543/UW96239 | 5752SCRRS | HBB7705 | Additional Modules for OW55775 |

| APAR/PTF Number | Product | FMID | Description |
|---|---|---|---|
| OW57544/UW96264 | 5752SCRRS | HBB7705 | Additional Modules for OW55775 |
| OW57545/UW96306 | 5752SCRRS | HBB7705 | Additional Modules for OW55775 |
| OW57561/UA01480 | 5752SCRRS | HBB7705 | Initialization Recovery Error |
| **z/OS 1.3** | | | |
| OA01584/UA01036 | 5752SCCTX | HBB7706 | 0C4 IN CTXRSWCTL |
| OA01876/UA01590 | 5752SCRRS | HBB7706 | Unexpected RC from ATRCMIT |
| OA01877, OA01879/ UA01608 | 5752SCRRS | HBB7706 | Unexpected Error Dumps during Restart / SPID Logging error |
| OA01916/UA02218/UA02219/UA022220 | 5695DF106 | HDZ11E0 HDZ11F0 HDZ11G0 | media manager returns twice for same call abend0c4 or other while preparing for or after second return. ow55469 applied. |
| OA02137/UA01597 | 5752SCRRS | HBB7706 | '0' output parms from atrbeg |
| OA02703/UA03795 | 5752SCXCF | HBB7706 | irlm delay. msgdxr168i msgdxr167e were issued due to delay in delivery by xcfas. lost usage of some of the mmdc mmdcs |
| OA04573/UA05763 | 5752SCRRS | HBB7706 | rrs latch contention for sys.atrbccdp |
| OA04771/UA06422 | 5752SCRRS | HBB7706 | contention showing deadlock for the she latch and the shat latch in atrxmscb |
| OA05073/UA08737 | 5752SCCTX | HBB7706 | latch for sys.atrurcpo not released when cntxinuse remains on not allowing the end context to complete |
| OA06191/UA08706 | 5752SCRRS | HBB7706 | task waiting on cntxinuse in atrbmecy leaving ur in forgotten (fgn) status. |
| OA08121 (same as OA07084 for higher releases) | 5752SCACB | HBB7706 | abend0c4 in isglrels because atbvsta attempted to release a garbage latch token. |
| OW50627/UW93831 | 5752SCRRS | HBB7706 | rollback rrs subordinate failure notify support |
| OW54120/UW93844 | 5752SCRRS | HBB7706 | rollback rrs subordinate failure notify support |
| OW55407/UW95848 | 5752SCRRS | HBB7706 | ctrace updates |
| OW55775/UW96160 | 5752SCRRS | HBB7706 | base reg usage / overlay error |

| APAR/PTF Number | Product | FMID | Description |
|---|---|---|---|
| OW56323/UA02634<br>OW56325/UA02634<br>OW56326/UA02634<br>OW56327/UA02634<br>OW56329/UA02573<br>OW56330/UA02573<br>OW56492 /UA02573<br>OW56526/UA02573<br>OA02517/UA02645<br>OA02518 /UA02657<br>OA02519/UA02668 | 5752SCRRS | HBB7706 | Misc Fixes for IMS reported problems |
| OW57349/UA02059 | 5752SCRRS | HBB7706 | RRS Hang while holding SYSZATR restart resource while rebuilds are occurring |
| OW57542/UW96213 | 5752SCRRS | HBB7706 | Additional Modules for OW55775 |
| OW57543/UW96240 | 5752SCRRS | HBB7706 | Additional Modules for OW55775 |
| OW57544/UW96265 | 5752SCRRS | HBB7706 | Additional Modules for OW55775 |
| OW57545/UW96307 | 5752SCRRS | HBB7706 | Additional Modules for OW55775 |
| OW57561/UA01481 | 5752SCRRS | HBB7706 | Initialization Recovery Error |
| **z/OS 1.4** | | | |
| OA01584/UA01037 | 5752SCCTX | HBB7707 | 0C4 IN CTXRSWCTL |
| OA01876/UA01591 | 5752SCRRS | HBB7707 | Unexpected RC from ATRCMIT |
| OA01877, OA01879/<br>UA01609 | 5752SCRRS | HBB7707 | Unexpected Error Dumps during Restart / SPID Logging error |
| OA01916/UA02218/UA02219/UA022220 | 5695DF106 | HDZ11E0<br>HDZ11F0<br>HDZ11G0 | Media manager returns twice for same call abend0c4 or other while preparing for or after second return. Ow55469 applied. |
| OA02137/UA01598 | 5752SCRRS | HBB7707 | '0' Output parms from ATRBEG |
| OA02703/UA03796 | 5752SCXCF | HBB7707 | Irlm delay. Msgdxr168i msgdxr167e were issued due to delay in Delivery by xcfas. Lost usage of some of the mmdc mmdcs |
| OA04573/UA05764 | 5752SCRRS | HBB7707 | Rrs latch contention for sys.atrbccdp |
| OA04771/UA06423 | 5752SCRRS | HBB7707 | Contention showing deadlock for the she latch and the shat latch in atrxmscb |
| OA05073/UA08738 | 5752SCCTX | HBB7707 | Latch for sys.atrurcpo not released when cntxinuse remains on Not allowing the end context to complete |

| APAR/PTF Number | Product | FMID | Description |
|---|---|---|---|
| OA06191/UA08707 | 5752SCRRS | HBB7707 | Task waiting on cntxinuse in atrbmecy leaving ur in forgotten (fgn) status. |
| OA07084/UA11534 | 5752SCACB | HBB7707 | ABEND0C4 in ISGLRELS because ATBVSTA attempted to release a garbage latch token. |
| OW55407/UW95849 | 5752SCRRS | HBB7707 | Ctrace Updates |
| OW55775/UW96161 | 5752SCRRS | HBB7707 | Base reg usage / overlay error |
| OW56323/UA02635<br>OW56325/UA02635<br>OW56326/UA02635<br>OW56327/UA02635<br>OW56329/UA02574<br>OW56330/UA02574<br>OW56492 /UA02574<br>OW56526/UA02574<br>OA02517/UA02646<br>OA02518 /UA02658<br>OA02519/UA02669 | 5752SCRRS | HBB7707 | Misc Fixes for IMS reported problems |
| OW57349/UA02060 | 5752SCRRS | HBB7707 | RRS Hang while holding SYSZATR restart resource while rebuilds are occurring |
| OW57542/UW96214 | 5752SCRRS | HBB7707 | Additional Modules for OW55775 |
| OW57543/UW96241 | 5752SCRRS | HBB7707 | Additional Modules for OW55775 |
| OW57544/UW96266 | 5752SCRRS | HBB7707 | Additional Modules for OW55775 |
| OW57545/UW96308 | 5752SCRRS | HBB7707 | Additional Modules for OW55775 |
| OW57561/UA01482 | 5752SCRRS | HBB7707 | Initialization recovery error |
| **z/OS 1.5** | | | |
| OA01916/UA02218/UA02219/UA022220 | 5695DF106 | HDZ11E0<br>HDZ11F0<br>HDZ11G0 | Media manager returns twice for same call abend0c4 or other while preparing for or after second return. Ow55469 applied. |
| OA04573/UA05765 | 5752SCRRS | HBB7708 | Rrs latch contention for sys.atrbccdp |
| OA04771/UA06424 | 5752SCRRS | HBB7708 | Contention showing deadlock for the she latch and the shat latch in atrxmscb |
| OA05073/UA08739 | 5752SCCTX | HBB7708 | Latch for sys.atrurcpo not released when cntxinuse remains on Not allowing the end context to complete |
| OA06191/UA08708 | 5752SCRRS | HBB7708 | Task waiting on cntxinuse in atrbmecy leaving ur in forgotten (fgn) status. |

| APAR/PTF Number | Product | FMID | Description |
|---|---|---|---|
| OA07084/UA11535 | 5752SCACB | HBB7708 | ABEND0C4 in ISGLRELS because ATBVSTA attempted to release a garbage latch token. |

**End of Document**